MASTER EN SCIENCES INFORMATIQUES

Imagerie médicale
étude et développement d'un système de télémicroscopie virtuelle

Jadoul, Cédric

Award date:
2005

Facultés Universitaires Notre-Dame de la Paix, Namur

Institut d'Informatique

Année Académique 2004 - 2005

Imagerie Médicale
-
Etude et développement d'un système de télémicroscopie virtuelle

Cédric Jadoul

Mémoire présenté en vue de l'obtention du grade de Maître en Informatique.

Résumé

Le domaine d'application de la télémédecine est très vaste. Il englobe une série de pratiques toutes aussi différentes les unes des autres mais dont l'objectif final est le même à savoir : la prestation à distance de soins médicaux. Parmi ces applications, la télémicroscopie est une discipline qui permet à un biologiste de visualiser à distance une lame porte-objets afin de rendre un diagnostic. La problématique d'une telle application réside dans la capacité à offrir aux biologistes, habitués à travailler avec un microscope conventionnel, un outils leur permettant de prester leur savoir-faire à distance de la manière la plus optimale possible.

Faisant suite à différents travaux déjà réalisés dans ce domaine, ce mémoire présente une solution d'application de visualisation de télémicroscopie appelée "microscope virtuel". Il réalise d'une part une analyse approfondie de l'état actuel de la recherche dans le domaine de la télémicroscopie, d'autre part, une étude en matière d'architecture débouchant sur une solution efficace et utilisable dans le domaine médical.

Mots-clefs : Télémicroscopie, Microscope Virtuel, JPEG2000, Méga image, Télémédecine, Télédiagnostique, Cytologie hématologie, Imagerie médicale, Navigation, Caching, Prefetching

Abstract

The domain of telemedicine is very vast. It includes a series of practices that differ from each other but whose final objective is the same : the remote service of medical care. Among these applications, the telemicroscopy is a discipline which gives the possibility to a biologist to visualize remotely a slide in order to give a diagnosis. The problems of such an application lie in the capacity to offer the biologists, who are accustomed to work with a conventional microscope, a tool allowing them to prest their know-how remotely in the most optimal possible way.

In accordance with various research already completed in this field, this thesis presents a solution of application of visualization of telemicroscopy called a "virtual microscope". The thesis is divided into two parts : On the one hand, it carries out a thorough analysis of the current state of research in the field of the telemicroscopy ; On the other hand, it carries out an architecture study leading to an efficient and usable solution in the medical field.

Keywords : Telemicroscopy, Virtual Microscop, JPEG2000, Mega image, Telemedicine, Telediagnostic, Cytology hematology, Medical imagery, Navigation, Caching, Prefetching

Remerciements :

Je tiens à remercier le Professeur Jean-Pol Leclercq pour avoir accepté d'être le promoteur de ce mémoire et pour ses relectures et son suivi lors de la réalisation de ce travail.

Je remercie également Monsieur Hubert Meurisse, co-promoteur de ce mémoire, pour ses judicieux conseils et sa disponibilité durant la réalisation de mon stage. Je lui suis également reconnaissant pour son aide quant à la structuration de ce mémoire et pour ses différentes relectures.

Je remercie également le Monsieur Bernard Chatelain et Monsieur Yvan Cornet, pour m'avoir aidé à dégager les objectifs de l'application à réaliser et pour avoir ensuite validé l'utilisation de celle-ci. Je les remercie également pour m'avoir conseillé dans la réalisation des parties médicales de ce mémoire.

Je voudrais aussi remercier Louis Zuyderhoff, ancien mémorant à Mont-Godinne, pour sa disponibilité et sa patience durant la réalisation de mon stage. Grâce à ses conseils et à son expérience du domaine, il a pu m'orienter et me permettre d'appréhender le terrain. Je le remercie également pour les lectures et critiques qu'il a apportées à la réalisation de ce mémoire.

Je remercie aussi Antonin Descampe pour sa disponibilité durant mon stage en m'aidant à comprendre les concepts fondamentaux du standard JPEG2000. Je tiens également à le remercier pour l'outil de création d'index qu'il a réalisé et adapté à mon application ainsi que pour ses critiques sur la partie JPEG2000 de ce mémoire.

Merci aussi à Guillaume Decock pour m'avoir expliqué le fonctionnement du microscope du laboratoire au début de mon stage et pour avoir répondu à mes questions à propos du standard JPEG2000.

Merci à Quentin Dallons et à Pierre Buyle pour m'avoir permis d'accéder au code source de leur mémoire ce qui m'a fortement aidé dans la compréhension et la mise en page de ce mémoire en LaTeX 2ε.

Je remercie tout particulièrement ma mère pour m'avoir toujours soutenu lors de mes études et pour sa patience lors des relectures de ce travail.

Finalement, je tiens à saluer tous mes collègues de classe pour les moments inoubliables que nous avons passés durant ces trois années de maîtrise.

Table des matières

Table des figures

Introduction

Depuis plusieurs années, l'évolution des réseaux et la mondialisation de l'Internet, ont entraîné une explosion des applications permettant de réaliser des traitements à distance dans des domaines plus variés les uns que les autres. La médecine est l'une de ces disciplines qui a su tirer profit de cette technologie émanante par l'introduction de la **télémédecine**. En effet, depuis l'établissement d'un diagnostic jusqu'à l'opération chirurgicale, tout acte médical peut aujourd'hui s'envisager à distance.

Ce travail propose de se concentrer plus particulièrement sur le domaine de la microscopie à distance dénommée plus précisément la **télémicroscopie**. Cette discipline offre la possibilité à un biologiste ou à un expert de visualiser à partir de son ordinateur, un échantillon (sang, tissus, liquide,...) se situant à n'importe quel endroit du monde. L'avantage d'une telle pratique réside dans l'accès à l'échantillon à distance. Il n'est en effet plus nécessaire d'envoyer l'échantillon. Le diagnostic est ainsi plus rapide.

C'est dans cette optique que j'ai réalisé mon stage de fin d'études au Laboratoire de cytologie des Cliniques Universitaires de Mont-Godinne. Récemment les travaux de L. Zuyderhoff (Cfr. [Zuy03]) ont permis la numérisation automatisée des images par l'intermédiaire d'un microscope à platine motorisée aboutissant à la réalisation d'échantillons de champs larges grâce à des techniques de corregistration spécifiques. Plus tard, les travaux de G. Decock (Cfr. [Dec04]), ont permis de stocker et de compresser ces images au format JPEG2000 sur un serveur dédié. C'est dans la continuité de ces travaux qu'il m'a été demandé de réaliser ce stage. Le but de ce travail était la réalisation d'une application de visualisation client-serveur permettant à un biologiste de consulter, grâce à son ordinateur, les mégaimages précédemment réalisées et ce, à la manière d'un microscope conventionnel.

Ce mémoire a donc pour objectif tant de présenter l'état actuel de la recherche en matière de télémicroscopie mais aussi la démarche qui a mené à la conception de l'application de visualisation.

La première partie dénommée "état de l'art" se compose de deux chapitres. Le premier chapitre a pour objectif d'énoncer le concept de télémédecine et de présenter l'état actuel de la recherche en télémicroscopie. Ce chapitre met l'accent sur les avantages offerts par la télémicroscopie mais prévient également d'une série d'inconvénients potentiels liés à l'utilisation de cette pratique. S'en suit une analyse de différentes solutions existant à travers le monde et une étude potentielle d'utilisations pour le laboratoire de cytologie. Le deuxième chapitre a pour principal objectif de faire la synthèse de l'état du système de télémicroscopie tel qu'il se présente actuellement au laboratoire de cytologie de l'UCL Mont-Godinne.

La deuxième partie est un approfondissement des différents outils utilisés pour la réalisation de l'application décrite dans ce mémoire. A cet effet, le premier chapitre est consacré à l'explication du standard JPEG2000 et aux avantages qu'il offre dans son utilisation pour une application de visualisation médicale. Le deuxième chapitre présente la technique du caching, énonce les avantages liés à son utilisation et décrit les différentes politiques de gestion d'une cache. S'en suit une explication de la méthode du préfetching.

La troisième partie est consacrée à la phase de réalisation de l'application. Le premier chapitre énonce l'ensemble des objectifs que l'application finale se devra de respecter pour être utilisable dans un domaine médical. S'en suit une série de choix qui nous permettront, lors du développement de l'application, d'être en phase avec les objectifs cités. Le deuxième chapitre présente l'architecture déployée pour la réalisation de l'application s'en suit une présentation générale de l'application finale.

Enfin, la quatrième partie joue une rôle critique dans la démarche de conception de l'application. A travers cette partie, nous tenterons de vérifier si l'application développée est en totale adéquation avec les objectifs fixés. Nous présenterons ensuite les limitations de l'application développée et nous proposerons finalement quelques pistes d'évolution.

Première partie

Etat de l'art

Chapitre 1

La télémicroscopie, la télémédecine au service de la microscopie

Ce chapitre est inspiré des références [RD], [MA], [CP02], [CDM+02] et de différentes informations recueillies auprès de Mr Chatelain B., Mr Meurrisse H. et Mr Cornet Y. des Cliniques Universitaires de Mont-Godinne.

1.1 Introduction

Depuis plusieurs années, avec l'évolution des technologies de l'information et de la communication (TIC), la télémédecine est devenue une pratique de plus en plus courante annonçant une profonde **mutation des usages de santé**.

'Depuis l'établissement d'un diagnostic[1] *jusqu'à l'opération chirurgicale, tout acte médical peut aujourd'hui s'envisager à distance. Les perspectives offertes par la télémédecine permettent d'imaginer que demain chacun pourra, partout et à tout moment, bénéficier des techniques médicales les plus pointues. Elles laissent également envisager une diminution des dépenses de santé (éviter des déplacements), l'accès aux soins par tous (y compris auprès des experts lorsque ceux-ci ne sont pas sur place), et une meilleure gestion des ressources (les experts sont rares).'*[RD]

Aujourd'hui, les projets de télémédecine fusent de toute part, tant dans les centres de recherche que dans des sociétés spécialisées (Cfr. Zeiss [Zei], Trestle [Tre], Scanscope [Sca]) ou non (Cfr. France Télécom [RD]). Pour beaucoup, la télémédecine est considérée comme '**la médecine de l'avenir**' et tous veulent proposer la solution la mieux adaptée qui remportera le

[1] Identification d'une maladie par ses symptômes *Cfr. Petit Larousse Illustré Edition 2005*

marché de demain. Pourtant, derrière cet enjeu économique, il en va de la vie de milliers de patients.

Dans ce chapitre, nous tenterons de définir clairement le concept de "**télémédecine**". Nous nous attarderons également sur un historique de la télémédecine, de sa naissance jusqu'à nos jours. Nous verrons également que le terme global "télémédecine" englobe beaucoup de pratiques. Nous en dresserons une typoligie. Dans le cadre de ce mémoire, nous concentrerons notre recherche sur le domaine de la **télémicroscopie** en donnant une définition précise et en en tirant les principaux avantages. Finalement, un état de l'art sur les applications les plus connues en télémicroscopie nous permettra d'évaluer l'intérêt de développer une nouvelle application de télémicroscopie pour le laboratoire de cytologie[2] de Mont-Godinne.

1.2 La télémédecine : définition

'La télémédecine est un concept général qui couvre différentes applications en rapport avec la santé. Elle constitue un domaine nouveau en plein développement qui s'appuie sur plusieurs technologies pour mettre en oeuvre des approches médicales nouvelles.'[CDM+02]

Selon l'Organisme mondiale de la santé (OMS), *la télémédecine est la pratique de soins médicaux utilisant des communications interactives sonores, visuelles et de données. Ceci inclut les prestations de soins médicaux et chirurgicaux, les consultations professionnelles, le diagnostic, ainsi que la formation et le transfert des données médicales.*

'Le plus grand avantage de la télémédecine réside dans l'accès aisé à des informations médicales n'importe où et n'importe quand. Le principe de base est de ***déplacer l'information et pas le patient****. La prise de décision pour lancer de nouveaux développements en télémédecine doit s'appuyer sur ce principe de base. C'est pourquoi les technologies de l'information et de la communication (TIC) jouent un rôle essentiel pour le transfert d'informations de manière sécurisée.'*[CDM+02]

En bref, ***'la télémédecine consiste en l'utilisation des télécommunications et des technologies de l'information pour permettre l'accès et la prestation des soins à distance et recueillir, organiser et partager les informations cliniques requises afin d'évaluer l'état du patient, de poser un diagnostic et d'établir un traitement.'***[MA]

1.3 Depuis sa naissance à nos jours

Le concept de télémédecine n'est pas récent. Il faut en effet remonter dans les années 50-60 en Amérique rurale (Texas et Géorgie) ainsi que dans

[2]Partie de la biologie qui étudie la structure et les fonctions de la cellule *Cfr. Petit Larousse Illustré Edition 2005*

le nord de la Norvège afin de découvrir les premières traces d'une ébauche de solution de télémédecine. A cette époque, la télémédecine permettait de répondre à des problèmes géographiques d'isolement de la population.

'Plus tard, suite à la multiplication des difficultés d'accès des patients à des centres de soins spécialisés, dans l'Amérique du début des années 80, plusieurs expériences ont été menées dans un souci de résoudre ces problèmes d'organisation. Il s'agissait d'expériences uniquement nationales et isolées.

Il faut pourtant attendre le début des années 90 pour que les sciences et technologies de l'information et de la communication (STIC) commencent à offrir les caractéristiques de débit, fiabilité, interopérabilité, coût d'investissement et de fonctionnement performants permettant l'utilisation sociale intensive que nous connaissons aujourd'hui (exemple de la Guerre du Golfe où plusieurs soldats de l'OTAN ont pu recevoir des soins depuis un centre allemand, via un système de visioconférence, de fax et de transfert de fichiers) dans le domaine de la santé publique où d'ailleurs les milieux hospitaliers ont largement contribué à leur essor.'[CP02]

Aujourd'hui, aux problèmes d'éloignements géographiques (tant des patients que des experts) s'ajoutent des problèmes économiques. Le secteur de la santé, dans presque tous les pays, est amené à répondre à deux demandes qui semblent être contradictoires : d'abord, fournir l'accès équitable aux services de santé de qualité et, ensuite, réduire ou contrôler les coûts croissants de services de santé. La télémédecine pourrait contribuer à satisfaire ces demandes en optimisant les utilisations des ressources existantes (telles que les spécialistes et le matériel) en se servant et en partageant de telles ressources par l'intermédiaire des liens de télécommunication. A l'heure actuelle, *'la télémédecine rencontre un intérêt considérable en Europe ainsi qu'au Canada et aux Etats-Unis. Le développement est d'autant plus prononcé dans les pays à grande extension géographique et à faible densité de population. D'autre part, les organisations humanitaires ont commencé à montrer la nécessité de recourir à la télémédecine pour pallier le manque de médecins dans les pays en voie de développement.'*[CDM$^+$02] De plus, le télémédecine permet de répondre au manque et à l'éloignement des experts car en effet, *'les experts sont rares'*[RD].

1.4 Typologie des applications de télémédecine

La télémédecine est, comme nous l'avons exprimé dans la définition 'un concept général qui couvre différentes applications en rapport avec la santé'. Il est, en effet, possible de distinguer plusieurs catégories d'applications en télémédecine. Dans l'étude [MPS$^+$], les auteurs en distinguent trois. Il est à noter que d'un auteur à l'autre, la dénomination des typologies peut varier. Le concept général reste cependant le même. Parmi les typologies des applications de télémédecine, on peut distinguer :

- **Téléconsultation** : Il s'agit ici d'évaluer le patient ou les données du patient à distance via un système de télécommunication. Le champ des applications de téléconsultation est très vaste. Il est encore possible de distinguer deux types de téléconsultation. Le **premier type** est celui où un patient consulte un médecin par une application de télécommunication interposée. Le **deuxième type** est celui dans lequel le médecin consulté sollicite un confrère à distance afin d'obtenir un deuxième avis diagnostique (télédiagnostic) et/ou thérapeutique (téléexpertise).
- **Télémonitoring ou télésurveillance** : Il s'agit ici, de surveiller, grâce à un appareillage spécifique, les paramètres physiologiques d'un patient à distance.
- **Téléformation ou télé-éducation** : *'Il s'agit d'un service de formation s'adressant à des étudiants ou à des professionnels de santé et par lequel ils peuvent avoir accès à un savoir-faire ou à des connaissances, quelle que soit leur localisation. On peut, par exemple, utiliser ce système pour réaliser une base de données médicale consultable par tous et à tout moment sur le web. Les internes en chirurgie peuvent assister à des opérations sans être physiquement présents dans la salle d'opération, ce qui résout certains problèmes d'hygiène et d'organisation de l'espace opératoire.'*[MPS$^+$]

Chaque typologie recense une multitude d'applications différentes. L'objectif de cet état de l'art étant de situer brièvement le concept de télémédecine, nous n'entrerons pas plus dans les détails de chacune des typologies. Le lecteur désireux d'en apprendre d'avantage peut consulter les sites et articles référencés en début de chapitre.

1.5 La télémicroscopie

Il faudrait plus qu'un sujet de mémoire pour traiter toutes les applications existant dans le domaine de la téléconsultation. Dans le cadre de ce mémoire, nous concentrerons notre état de l'art sur la télémicroscopie.

1.5.1 Définition

Intuitivement, on peut dire que '*la télémicroscopie est l'application de la télémédecine à la microscopie*' [Zuy03]. De manière plus précise, la télémicroscopie est une pratique de télédiagnostic ou de téléexpertise permettant à un spécialiste de visualiser sur un écran d'ordinateur (le poste client) une lame de microscope pouvant se trouver à des milliers de kilomètres (le poste serveur) de l'endroit où il se trouve dans le but de rendre un diagnostic. A cette fin, un moyen de communication doit exister entre les deux postes. Cette visualisation s'accompagne des outils traditionnels disponibles sur un microscope conventionnel (zoom, déplacement, focus). Des fonctionnalités

supplémentaires, qui n'existent pas sur le microscope conventionnel peuvent être ajoutées améliorant ainsi le confort du diagnostic (filtres de couleur, captures d'écran, annotations, etc.).

'Le but recherché est une amélioration du processus de décision médicale (et en conséquence, de la qualité de la prise en charge du patient) : par une plus grande collégialité de cette décision, par une meilleure coordination entre différentes décisions, par la mise à disposition de la compétence et de l'expertise d'un référent très spécialisé, etc.'[MPS$^+$]

Dans les mémoires [Geo02] et [Zuy03], les auteurs distinguent deux approches en télémicroscopie :

L'approche store and forward (ou statique) est un système qui consiste à sélectionner et numériser un échantillonnage représentatif du contenu de la lame afin de créer une lame virtuelle et la rendre disponible sur un serveur d'images. L'avantage principal de cette approche est de permettre des analyses 'asynchrones'. Cependant, cette approche se bute au problème crucial de l'échantillonnage[3]. C'est pourquoi, la résolution de la problématique de la numérisation de champs larges[4] est une priorité dans un tel système.

L'approche real time (ou dynamique) permet à un spécialiste de piloter un microscope à distance en temps réel. Un inconvénient est qu'il n'est plus possible d'effectuer d'analyses asynchrones. Une analyse à distance requiert en effet la présence d'une personne pour préparer les lames et la disponibilité de la station d'acquisition. De plus le coût de la mise en place d'une telle solution dépasse largement celui des solutions 'Store and forward'.

1.5.2 Cadre d'utilisation

Comme nous l'avons signalé dans la définition, la télémicroscopie s'inscrit dans le cadre d'une application de télémédecine orientée télédiagnostic ou téléexpertise. Afin de comprendre cette démarche, il peut être intéressant d'expliquer la démarche d'analyse d'une lame.

'Suite à la demande d'une analyse, le spécialiste utilise le microscope afin de scanner avec différents grossissements une lame de verre à la recherche d'anomalies. Lorsqu'une anomalie est détectée, elle est confrontée avec une série d'autres informations cliniques disponibles telles que le dossier médical du patient dans le but d'identifier un ou plusieurs diagnostics possibles.

Parfois, la difficulté du cas nécessite un deuxième avis : il peut s'agir de l'avis d'un collègue travaillant dans le même département, mais bien souvent l'avis d'une personne externe et experte dans le type de cas analysé est requis. Dans cette dernière situation, le spécialiste qui a effectué l'analyse doit communiquer à l'expert le support nécessaire à sa consultation.'[Zuy03]

[3]Partie de la lame qui sera archivée sous forme de lame virtuelle

[4]Dont le problème avait été soulevé dans [Zuy03] et une ébauche de solution avait été réalisée dans [Dec04]

'Dans certaines circonstances, le spécialiste belge ne peut pas se déplacer pour rencontrer son homologue étranger. Il n'est également pas possible d'envoyer un courrier contenant le prélèvement, car le délai de réponse serait trop important. Le téléphone reste la solution la plus envisageable. Dans ce cas, comment rendre un avis correct sans disposer d'un support visuel.'[Geo02] D'où l'utilité de l'utilisation de la télémicroscopie.

Bien que le cadre d'utilisation de la télémicroscopie soit en général la demande de l'avis d'un expert, celle-ci peut également apporter beaucoup d'avantages dans d'autres cadres d'utilisation : contrôles de qualité, formation et apprentissage à distance, etc. Nous n'entrerons cependant pas plus dans les détails.

1.5.3 Les avantages d'une application de télémicroscopie

La télémicroscopie étant une technique assez récente, son utilisation est encore assez restreinte. Peu de laboratoires peuvent à l'heure actuelle se vanter d'avoir un système de télémicroscopie au point. C'est pourquoi, il est impossible de se prononcer définitivement sur les avantages de la télémicroscopie avant que des recherches plus approfondies ne soient menées pour déterminer la valeur de cette technique en ce qui concerne son efficacité et son rendement. Les avantages qui sont cités ci-dessous sont donc issus d'informations relevées au cours de la réalisation de ce mémoire et ne concernent donc qu'un petit échantillon de personnes. Toutefois, on s'entend généralement sur le fait que la télémicroscopie inscrite dans le cadre d'utilisation globale d'une application de télémédecine comporte des possibilités importantes et qu'elle est susceptible d'apporter de nombreux avantages pour les patients, les médecins, les spécialistes, les soins de santé et l'hôpital en lui-même.

Avantages pour le laboratoire

- **De meilleures relations avec les autres laboratoires et médecins.** L'utilisation d'une application de télémicroscopie peut permettre de resserrer les liens entres les laboratoires qui pourront ainsi travailler en plus étroite collaboration.
- **Des prestations plus efficaces (moins coûteuses et plus rapides)** Les prestations sont bien sûr plus rapides : aucun délai n'est nécessaire à l'envoi de la lame. Celle-ci est accessible instantanément par le spécialiste à distance. Cette façon de procéder permet également de limiter les frais d'envois, de réponse,.... Il en va donc de la notoriété du labo.
- **Les contrôles de qualité** permettent à un laboratoire de tester les qualités d'analyses d'autres laboratoires. Grâce à l'utilisation d'une application de télémicroscopie, il n'est plus nécessaire d'orienter l'échan-

tillonnage (mettant en évidence les cellules à analyser). Les laboratoires testés devront fouiller la totalité de la lame porte-objet proposée et fournir un diagnostic. La télémicroscopie permet donc un meilleur contrôle de ces laboratoires. Et, non négligeable, la rapidité de la mise à disposition d'une lame porte-objet par le biais de ce type d'application permet de réaliser des gains de temps importants.
- **L'archivage** des lames porte-objet peut être géré de manière plus efficace. Cet archivage est réalisé afin de permettre une analyse rétrospective ou même pour l'apprentissage (archivage de cas rares). Cependant, les lames porte-objet traditionnelles sont fragiles. En effet, la coloration peut s'altérer avec le temps. De plus, il est impossible de les reproduire. Les lames numérisées quant à elles offrent une durée de vie presque illimitée ainsi que la possibilité de les reproduire autant de fois qu'on le souhaite.

Avantages pour les médecins, biologistes,...

- **La formation** est un aspect très important. L'archivage des maladies déjà rencontrées, des cas rares,... permet de contribuer à une bonne formation du personnel. De plus, la télémicroscopie permet à plusieurs biologistes se trouvant aux quatre coins du monde de visualiser ensemble une "situation", de l'analyser et de partager leur point de vue.
- **Les meilleurs recommandations** fournies par le biologiste. Les experts étant plus facilement accessibles, le biologiste pourra rendre le diagnostic le plus juste en cas de doute.
- **Un meilleur confort de visualisation** grâce à la possibilité de manipuler l'image en travaillant sur les niveaux de couleurs, la luminosité, le contraste, etc.

Avantages pour le patient

- **La prévention des complications** grâce à la prestation plus rapide de meilleurs soins. Dans l'urgence, il n'est pas toujours possible de faire appel à un expert à cause des délais trop longs dus à l'envoi de la lame porte-objet ou du déplacement de l'expert. Grâce à la télémicroscopie, la disponibilité quasi immédiate de la lame numérisée aux quatre coins du monde, permet de demander d'urgence un avis médical à un expert et donc de prodiguer les meilleurs soins au plus vite.

Avantages économiques

- **La réorganisation** du laboratoire par l'introduction d'un outil *'d'automatisation d'une analyse permet de libérer un spécialiste pouvant dès lors être affecté à une autre tâche et permet entre autre une diminution de la durée d'analyse.'*[Geo02]

– **La possibilité de facturation** de l'expertise prestée par un spécialiste de l'hôpital est également un avantage économique. En effet, à l'heure actuelle, les hôpitaux ne sont pas toujours rémunérés pour la prestation d'une telle expertise. Grâce à ce type d'application, il peut être envisageable d'ajouter un système de facturation automatique lorsqu'une prestation est réalisée.

1.5.4 Quelques inconvénients potentiels

L'utilisation d'une application de télémicroscopie peut présenter certains inconvénients. Toutefois, le terme 'inconvénient' n'est pas nécessairement bien choisi. En effet, certains problèmes pourront être contournés grâce à des techniques existantes ou futures, des formations, etc. C'est pourquoi, nous nous entendrons sur le fait que les inconvénients cités ci-dessous sont plus des mises en garde plutôt que de réels problèmes liés à l'utilisation d'une application de télémicroscopie.

Inconvénients techniques

– **Les exigences visuelles** des biologistes peuvent ne pas être satisfaites par les images proposées par l'application (perte de qualité, mauvais rendu de couleurs, etc).
– **Des problèmes de lenteur** dus à une surcharge réseau par exemple peuvent altérer le confort de visualisation se traduisant parfois par un moins bon diagnostic.
– **Des bugs** possibles peuvent également porter préjudice lors d'une intervention urgente.

Inconvénients pour les médecins, biologistes, ...

– **L'adaptation des biologistes** à la nouvelle technologie est également un facteur important. En effet, une application de télémicroscopie est un changement assez radical dans le mode de travail d'un biologiste. Des difficultés d'adaptation peuvent alors apparaître dues par exemple à l'absence de contact physique avec le microscope réel ou la lame porte-objets.

Inconvénients économiques

– **Des coûts importants** engendrés par l'achat de matériel et de software pour l'installation d'une station de télémicroscopie dans un laboratoire peuvent ne jamais être rentabilisés. En effet, certains biologistes pourraient ne pas s'adapter à l'application et continuer à vouloir travailler de manière traditionnelle laissant ainsi l'infrastructure totalement à l'abandon.

Inconvénients pour le patient

– **Des coûts de prestations** revus à la hausse par les laboratoires dans le but de rentabiliser au plus vite la nouvelle infrastructure de télémicroscopie.

Tout comme nous l'avons signalé pour les avantages, nous pourrons convenir plus précisément des inconvénients liés à l'utilisation de la télémicroscopie lorsque celle-ci sera utilisée de manière plus intensive dans les laboratoires.

1.5.5 Quelques applications de télémicroscopie

Jusqu'à présent, nous avons défini le concept de télémicroscopie et nous en avons cité les principaux avantages. Afin de compléter cet état de l'art, il peut être intéressant de citer quelques applications connues en télémicroscopie. En effet, certains centres de recherche et certaines sociétés ont développé des systèmes offrant la possibilité à un biologiste de visualiser une lame à distance.

Dans cette optique, certaines applications se basent tantôt sur une approche "real time" c'est-à-dire permettant un pilotage du microscope à distance tantôt sur une approche "store and forward" proposant à l'utilisateur la manipulation d'une lame virtuelle[5] stockée sur un serveur d'images.

Avant toute chose, notons qu'une "bonne" application de télémicroscopie doit au moins proposer les mêmes outils qu'un microscope conventionnel (déplacement X-Y, vitesse de déplacement, changement d'objectif, focus). Nous verrons qu'en plus des fonctionnalités de base, la plupart des applications ajoutent d'autres fonctionnalités permettant une visualisation et un diagnostic plus souple et plus rapide.

Dans ce point, nous allons donc donner un descriptif d'applications connues dans le monde de la télémicroscopie. Cette liste n'a pas la prétention de recenser l'entièreté des applications existantes dans le domaine mais elle représente un échantillonnage précieux de diverses solutions envisageables.

Zeiss : Le système Axiopath - Remote vision with a mouse click

Zeiss propose un système de télémicroscopie se basant sur une approche "real time". Il est, en effet, possible grâce à un appareillage spécifique, de contrôler le microscope à distance.

La figure 1.1 illustre l'architecture mise en place pour cette application.

Cette architecture est divisée en deux parties : la partie client et la partie serveur. Ces deux entités sont reliées à l'aide d'un réseau de communication quelconque (satellite, LAN, ISDN, ADSL, etc.).

[5] C'est-à-dire une lame ayant été numérisée à l'aide d'un appareillage spécifique (scanner ou microscope)

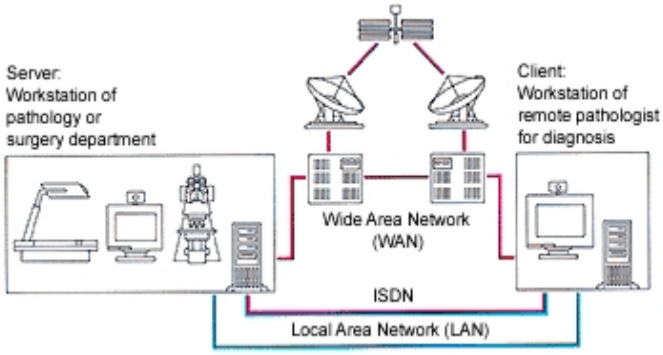

FIG. 1.1 – Zeiss - Axiopath - Architecture [Zei]

Le côté client est la station de visualisation composée d'un ordinateur relié à un réseau de communication et sur lequel est installé le logiciel Axiopath permettant de manipuler le microscope à distance.

Le côté serveur est la station d'acquisition. Cette station se compose d'un ordinateur relié à un réseau de communication et sur lequel le logiciel Axiopath est également installé. Ce même ordinateur est relié à un microscope à qui il transmet les ordres de déplacement, changement d'objectif, focus. Une caméra fixée sur le microscope et reliée à l'ordinateur permet de transmettre les images du microscope en temps réel.

La figure 1.2 illustre l'application de visualisation en mode "real time".

De manière simple, en mode "real time", l'utilisateur se connecte à un microscope distant disposant de la lame à analyser. L'utilisateur reçoit immédiatement une 'slide view'[6] lui permettant de choisir la zone qu'il désire visualiser. Lorsque cette zone est choisie, l'utilisateur clique sur "Scan Slide Region" envoyant ainsi l'ordre au microscope de se positionner sur les coordonnées X-Y déterminées et de scanner cette région. L'image scannée apparaît alors dans la zone principale.

L'application propose également de se déplacer dans la lame de la même manière qu'avec un microscope conventionnel (déplacement axe X-Y, vitesse de déplacement, changement d'objectif).

Parmi les fonctionnalités avancées, on note la possibilité de réaliser des annotations directement sur l'image, de réaliser des captures d'écran, la vidéo conférence.

Notons simplement que l'application propose également une approche "store and forward" limitée. L'utilisateur a la possibilité de travailler sur des captures d'images archivées et non sur l'entièreté de la lame scannée. Celui-ci n'a donc plus la possibilité de voyager dans celle-ci comme on le ferait

[6]Entièreté de la lame porte-objets sous forme d'une imagette

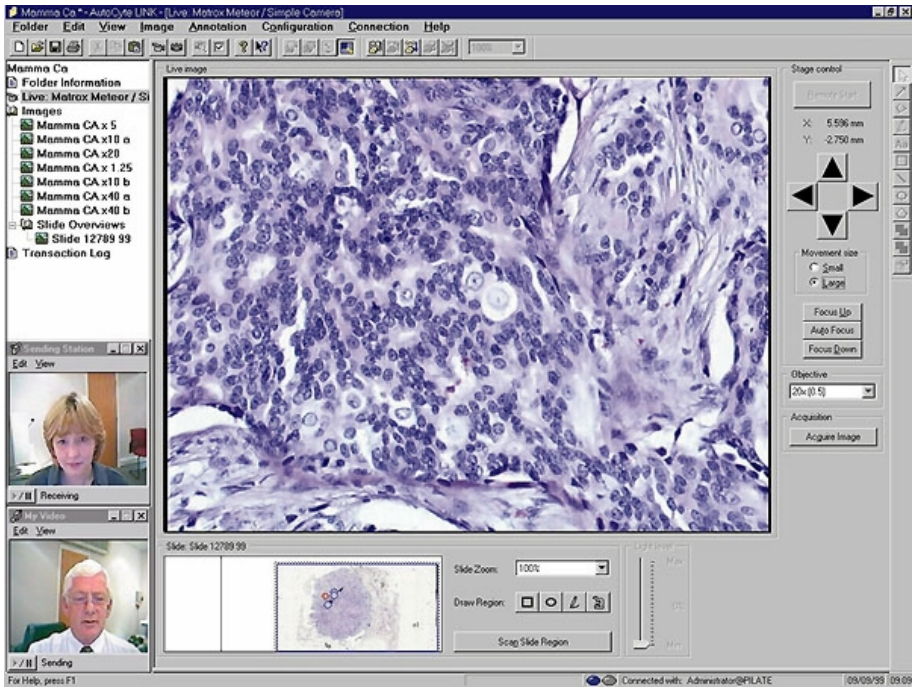

FIG. 1.2 – Zeiss - Axiopath - Remote vision with a mouse click [Zei]

pour une application de télémicroscopie ”store and forward ” digne de ce nom.

En résumé, l'application offre une qualité d'interface et des fonctionnalités très intéressantes en mode ”real time”. Cependant, des fonctionnalités manquent à l'appel (luminosité, contraste, niveaux RGB). Le mode ”store and forward” ne permet pas quant à lui de réaliser tout ce que l'on peut attendre d'une application de télémicroscopie ”store and forward”.

Trestle Corporation - MedMicroscopy - Internet Microscopy Solution

Trestle propose également un système de télémicroscopie ”real time”. L'architecture de l'application est la même que celle proposée par Axiopath. L'avantage par rapport à Axiopath est que Trestle propose de connecter, si possible, l'équipement que le laboratoire possède et non d'en vendre un nouveau.

'MedMicroscopy uses standard PCs to connect microscopes to the Internet, allowing users to view and navigate slides from accross town, accross the country, or around the world..... Because the system works with existing equipment over standard Internet connections, MedMicroscopy requires minimal hardware and no fixed data lines, making it a cost-effective invest-

ment for corporations, universities, research labs, and hospitals seeking a more efficient use of microscopes.' [Tre]

La figure 1.3 illustre l'application MedMicroscopy côté client.

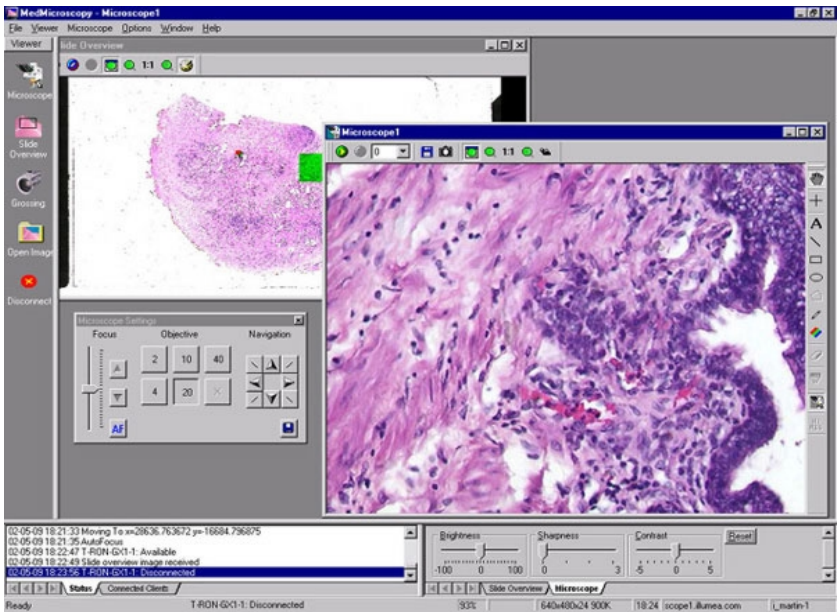

FIG. 1.3 – Trestle Corporation - MedMicroscopy [Tre]

Outre les fonctionnalités de base (déplacement XY, zoom, etc.), l'application MedMicroscopy répond à l'une des critiques faite à Axiopath. Il est en effet ici possible de travailler sur les niveaux de contraste, de luminosité et de brightness.

L'application se distingue également par la possibilité de réaliser des mesures directement sur l'image et d'effectuer des captures d'écran sous format JPG, TIF ou BMP.

En résumé, l'application proposée par Trestle est très complète au niveau des fonctionnalités proposées. De plus, son faible coût d'investissement en fait une application très intéressante pour une approche "real time". Cependant, on peut reprocher à l'application de ne pas s'orienter vers l'approche "store and forward" qui pourrait s'avérer très utile dans certains domaines tels que la cytologie.

Scanscope - Welcome to the world of virtual slide technology

Scanscope s'oriente vers une approche "store and forward". En effet, la société offre la possibilité de numériser des lames porte-objets et de les rendre accessibles en utilisant le browser web. L'avantage de ce type d'application est qu'il n'est pas nécessaire d'investir dans du matériel d'acquisition.

'Let us convert your glass microscope slides into high-resolution, Internet-accessible digital slides for you.'[Sca]

Comme le montre la figure 1.4, le processus se passe en plusieurs étapes.

FIG. 1.4 – Etapes scanscope [Sca]

Le processus se décompose en trois grandes étapes :

- Etape 1 : Le laboratoire envoie ses lames porte-objets à Scanscope. A l'aide d'un scanner, ScanScope transforme ces lames en lames virtuelles. Les grossissements proposés sont 20x et 40x.
- Etape 2 : Une fois les lames scannées, un CD ou un DVD contenant les slides virtuels est envoyé au laboratoire. De plus, ces slides sont mis à disposition sur Internet durant 12 mois.
- Etape 3 : Via son browser web ou une application viewer téléchargeable gratuitement sur le site de Scanscope, il est possible à l'utilisateur de consulter, d'annoter les lames virtuelles à partir de n'importe quel endroit.

La figure 1.5 illustre la visualisation proposée par scanscope à partir du browser web.

L'utilisateur a la possibilité de réaliser les mêmes fonctionnalités qu'avec un microscope conventionnel(zoom, déplacement XY, etc.). Il est à noter que la fonctionnalité "slide view" est également disponible ce qui est un plus par rapport au microscope conventionnel.

La figure 1.6 illustre l'application viewer proposée par scanscope.

Outre les fonctionnalités classiques du microscope conventionnel, l'application propose également la possibilité de réaliser des captures d'écran, des mesures sur l'échantillon et des annotations.

En résumé, scanscope offre un procédé complet de télémicroscopie "store and forward" à un coût très réduit. Au niveau des deux applications, on peut reprocher l'absence de fonctionnalité permettant un travail sur les couleurs et la luminosité de l'image. Au niveau technique, les grossissements proposés peuvent ne pas satisfaire certains domaines tels que la cytologie qui demande un niveau de grossissement encore plus élevé pour permettre un bon diagnostic.

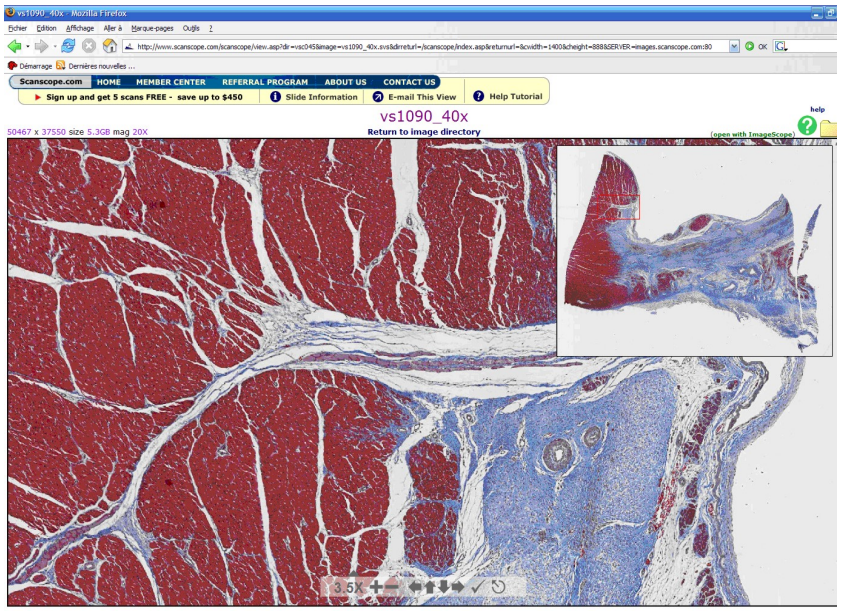

FIG. 1.5 – Scanscope - Application Web [Sca]

1.5.6 Besoins du laboratoire de Cytologie

Dans la section précédente, nous avons présenté trois applications réputées dans le domaine de la télémicroscopie. A la lumière de ces différentes applications, il serait intéressant d'étudier la possibilité d'en utiliser une d'entre elles pour le laboratoire de cytologie des Cliniques Universitaires de Mont-Godinne.

Dans la définition de la télémicroscopie, nous avons mis en évidence deux types d'approche (”real time” ou ”store and forward”). On peut se demander quelle approche conviendrait le mieux au domaine de la cytologie. De toute évidence, l'approche ”store and forward” présente des avantages beaucoup plus intéressants que l'approche ”real time”. En effet, des avantages tels que l'archivage électronique, la non péremption des lames numérisées, la non nécessité de présence d'un technicien lorsque quelqu'un désire visualiser la lame à distance, le coût moins élevé, nous font pencher pour ce type de solution. Nous n'affirmons cependant pas qu'une application ”real time” n'est pas envisageable. Cependant, comme le dit le Dr Joel de l'université d'Oxford en parlant de l'approche ”real time” : *'You'll need a fat wallet and a sharp mind to play ball here.'*[Leo]. De plus, les avantages offerts par une approche ”store and forward” s'avèrent beaucoup plus pertinents. C'est pourquoi, nous décidons d'orienter notre analyse sur les applications ”store and forward”.

Lors de nos différentes recherches, nous avons pu noter qu'il existait un

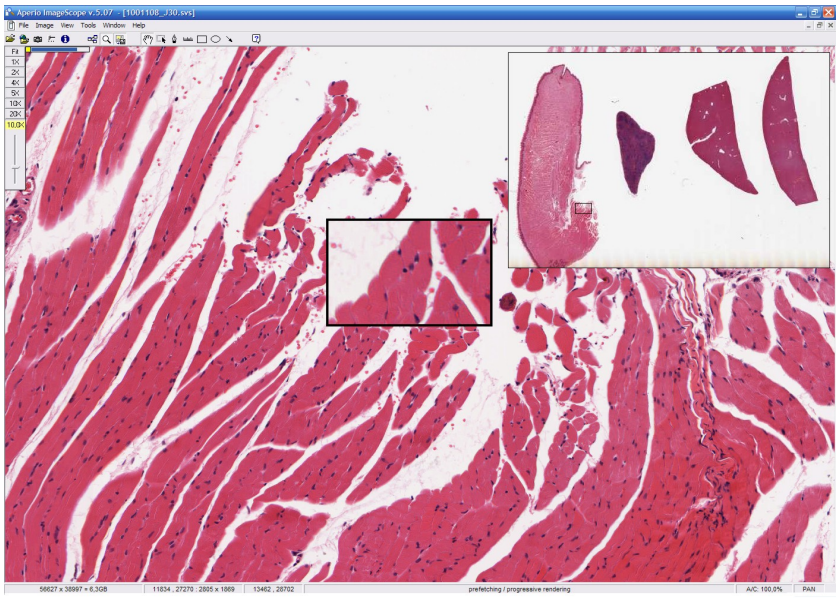

FIG. 1.6 – Scanscope - viewer [Sca]

grand nombre d'applications de télémicroscopie "real time". Les applications "store and forward" quant à elles étaient beaucoup moins nombreuses. Mis à part la solution proposée par scanscope, il n'existe à l'heure actuelle, pas de solution fiable offrant cette approche. De cette constatation on est en droit de se demander quelles sont les raisons de ce retard en matière de développement d'applications de télémicroscopie "store and forward" ?

Dans un domaine comme celui de la cytologie, le travail du cytologiste consiste en la fouille d'un échantillon afin de détecter d'éventuelles anomalies correspondant à une pathologie. C'est pourquoi, il est impératif que la numérisation de la lame porte-objets soit la plus représentative possible. Dans cette optique, la problématique d'une application "store and forward" réside dans la difficulté de numériser des grands champs appelés "méga-images" par mosaïquage de champs conjoints. Le problème ayant déjà été évoqué dans d'autres mémoires, nous n'entrerons pas plus dans les détails. Cependant, le lecteur désireux d'en apprendre plus, peut consulter les ouvrages [Zuy03] et [Dec04].

La solution "store and forward" proposée par scanscope permet d'éviter le problème de mosaïquage de champs conjoints à l'aide d'un scanner. Cependant, si ce type de numérisation donne des résultats satisfaisants dans le domaine de l'histologie[7], ceux-ci sont loin de satisfaire aux exigences des

[7]Spécialité médicale et biologique qui étudie au microscope la structure des tissus des êtres vivants *Cfr. Petit Larousse Illustré Edition 2005*

cytologistes. Afin de démontrer ces propos, nous avons demandé à Scanscope de numériser une lame porte-objets que nous leur avons envoyée.

La figure 1.7 illustre la comparaison entre la numérisation proposée par Scanscope et celle attendue par un cytologiste. La capture de Scanscope a été réalisée à l'aide d'un scanner au grossissement 40x[8]. La capture de cytologie, a été réalisée au grossissement 100x à l'aide d'un microscope et d'une caméra numérique présente au laboratoire de cytologie de Mont-Godinne.

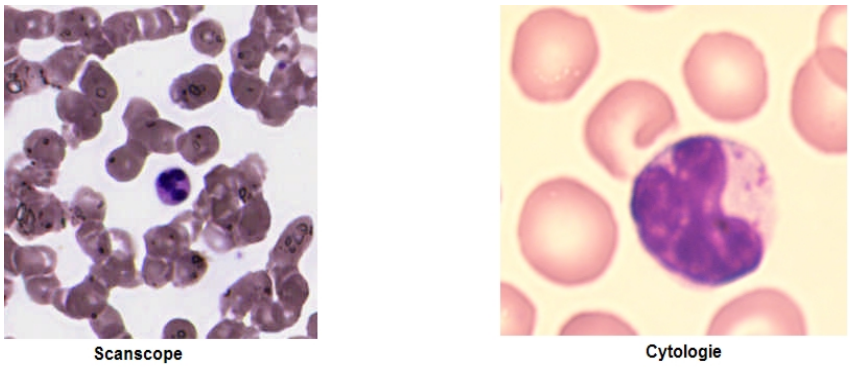

FIG. 1.7 – Comparaison image Scanscope et Cytologie

La comparaison est éloquente. Deux problèmes majeurs apparaissent clairement. La première faiblesse se situe au niveau du grossissement. Scanscope propose un niveau d'optique de 40x alors que les besoins de la cytologie se situent vers un niveau de grossissement de 100x. Le deuxième handicap est sans conteste le rendu des couleurs obtenu avec le scanner. Il est quasi impossible sur l'image de Scanscope de distinguer les types de cellules.

Cette application ne peut dont être retenue pour le laboratoire de cytologie de Mont-Godinne. Il est à noter également que le laboratoire de Mont-Godine dispose déjà d'une station d'acquisition. Il serait donc dommage de ne pas utiliser ce dans quoi on a investi.

1.6 Conclusion

Cette analyse nous fait donc prendre conscience qu'il n'existe, à l'heure actuelle, aucune application de télémicroscopie "store and forward" répondant totalement aux besoins des cytologistes. De plus, l'existence d'une station d'acquisition ainsi que différents travaux réalisés au sein du laboratoire de cytologie de Mont-Godine dans cette optique (cfr. Chapitre 2) nous font pencher vers la nécessité de réalisation d'une application de télémicroscopie "store and forward" personnalisée pour le laboratoire de cytologie de

[8]Grossissement maximum autorisé par le scanner

Mont-Godinne.

Cette démarche permettra de fournir un produit correspondant aux besoins d'exploitation des procédures microscopiques spécifiques au laboratoire de Mont-Godine en vue d'améliorer les conditions du diagnostic en combinant certains outils intéressants relevés dans l'analyse des applications existantes (capture d'écran, slide view, mesures, etc.).

L'avis du Dr F. Joel Leong de l'Université d'Oxford, spécialiste en imagerie médicale conforte également cette décision : *'Get into telepathology the cheap way. All you need is a digital capture device and an internet connection. If you have money to burn, you will purchase specialised software - this is completely unnecessary .... Do not discount the value of store-and-forward telepathology. With sophisticated software interface design and in the context of a digital slide, it is possible to simulate an interactive experience and fool many pathologists into believing they are actually using a microscope in real-time. Store-and-forward has a bigger future than you realise...'* [Leo].

Chapitre 2

La télémicroscopie aux Cliniques Universitaires de Mont-Godinne

2.1 Introduction

Dans le chapitre précédent, nous avons tenté de comprendre pourquoi la réalisation d'une application de télémicroscopie pour le laboratoire de cytologie de Mont-Godinne était une nécessité. A travers cette recherche, nous avons pu constater que le problème de coregistration était confronté à la prise en compte de "très grands champs" et qu'à l'heure actuelle, il n'existait pas de solution concrète à ce problème.

Depuis plusieurs années, des mémorants se sont succédé au sein du laboratoire de cytologie de Mont-Godinne afin de rélever ce défi. Comme nous pourrons le constater, ceci ne s'est pas fait sans mal. Bien qu'étant dans un état de réalisation avancée, l'application ne peut encore, à l'heure actuelle, être utilisée de manière optimale. En effet, il subsiste certains points à revoir, à améliorer ou même à développer complètement.

Dans ce chapitre, nous tenterons d'identifier les différents travaux ayant participés à la réalisation d'une partie de l'application de télémicroscopie présente actuellement au sein laboratoire de cytologie. Les modules posant encore certains problèmes ou restant à développer seront également identifiés[1]. Notons que les différents travaux cités seront décrits dans les grandes lignes. Le lecteur désireux d'en apprendre davantage sur les détails des différentes solutions citées est invité à consulter les ouvrages référencés[2].

[1] Les problèmes identifiés sont issus de l'état de l'art de l'application telle qu'elle était avant la réalisation de ce présent travail

[2] Il s'agit de [Geo02], [Zuy03], [Dec04]

2.2 Etat de l'application de télémicroscopie du laboratoire de cytologie

Une application de télémicroscopie ne se limite pas seulement à un outil de visualisation. La visualisation n'est en effet que le dernier élément d'une longue série d'étapes ayant permis de réaliser une lame virtuelle grâce à la numérisation d'une lame porte-objets. La figure 2.1 tirée de [Dec04] illustre les différents modules nécessaires à la réalisation d'une application de télémicroscopie de type "store and forward".

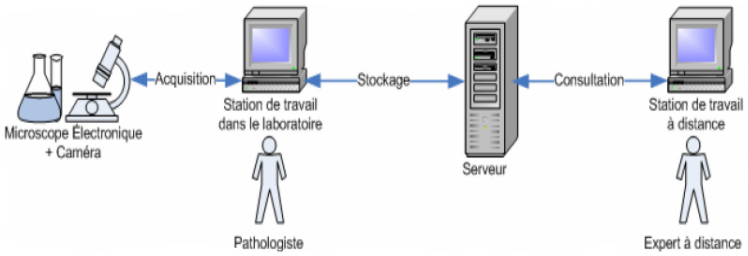

FIG. 2.1 – Schéma général d'une application de télémicroscopie de type "store and forward"

Ce schéma laisse apparaître trois modules majeurs : l'acquisition, le stockage et la visualisation. Dans les points suivants, nous allons identifier les mécanismes mis en place pour la réalisation de chacun de ces modules, les difficultés rencontrées, la solution proposée par les précédents mémorants et les problèmes persistants.

2.2.1 L'acquisition

Cette étape a pour but de numériser une lame porte-objets à l'aide d'un microscope à platine motorisée et d'une caméra digitale. C'est cette étape qui va permettre de transformer "l'état physique" de la lame en un "état virtuel", manipulable à l'aide d'un outil informatique. Dans ce processus d'acquisition, il convient d'identifier deux sous processus à savoir **la capture d'images** et **la coregistration** de ces mêmes images. Précisons que ce sous processus de coregistration est lié à la création de mégaimages et non à un processus typique d'une application de télémicroscopie store and forward.

La capture des images

Cette étape qui fait partie intégrante du processus d'acquisition consiste à capturer des galeries d'images représentant la zone à numériser. Au laboratoire de cytologie de Mont-Godinne, ce procédé est réalisé grâce au microscope Olympus Ax70 et à une caméra analogique Sony DXC-950 (voir figure

FIG. 2.2 – Microscope Olympus "Ax70" et Caméra Analogique Sony "DXC-950"

2.2) fixée au-dessus de celui-ci. Cette même caméra, reliée à un ordinateur permet de capturer une série d'images qui seront fournies au processus de coregistration pour la réalisation d'une image grand champs.

Afin de réaliser des captures, l'ordinateur relié au microscope est doté d'un programme "Analysis Pro 3.0". *'Ce logiciel est conçu pour l'acquisition, l'analyse, le traitement et l'archivage d'images. Il présente plusieurs avantages. D'une part il est doté d'un module comprenant des librairies permettant de gérer les différentes fonctionnalités du microscope « AX70 » d'Olympus, d'autre part il possède un composant permettant à l'utilisateur de personnaliser l'application en créant lui-même ses modules personnels. Le langage utilisé par ce composant est un langage proche du ANSI-C appelé "Imagin-C" et qui a comme particularité de proposer certaines fonctions supplémentaires liées au traitement d'images, ainsi que de proposer un interpréteur intégré. Imagin-C est également compatible avec le "MS Windows Sorfware Developement Kit " (SDK) ce qui permet la programmation de nouvelles interfaces graphiques dans l'environnement "MS Windows". Le logiciel Analysis permet aussi une intégration facile des nouvelles fonctions dans son interface, en y associant des boutons.'*[Dec04]

La difficulté de ce processus réside donc dans la compréhension et l'utilisation du logiciel analysis et du langage Imagin-C. Dans son ouvrage [Geo02], Benoît Georges décrit un procédé permettant de réaliser automatiquement des galleries d'images en utilisant les modules proposés par le logiciel Analysis. Cette approche de télémicroscopie consiste donc à orienter le diagnostic en proposant directement une galerie de cellules à analyser. Cependant, la volonté de capturer des grands champs s'est rapidement fait ressentir car comme le signale Mr Yvan Cornet du laboratoire de cytologie :*'Si la galerie d'images sert à rassembler les globules blancs en une seule vision (pour un comptage ou une comparaison cellule à cellule), la mégaimage sert à avoir une vue d'ensemble du frottis pour une appréciation globale de la richesse, de la distribution des éléments, de la morphologie des globules rouges, des plaquettes (éléments qui sont difficiles à apprécier cellule par cellule). En bref,*

la galerie a été conçue à la base pour effectuer une formule sanguine (globules blancs) de manière rapide et pour détecter les globules blancs anormaux en un seul coup d'oeil. La mégaimage a été pensée pour les prélèvements médullaires[3] *où la richesse cellulaire est plus importante et où la mégaimage est plus efficace. L'idée d'une mégaimage nous paraissait plus adéquate pour restituer l'entièreté du frottis sanguin apportant ainsi les sensations de posséder l'entièreté de l'échatillonnage (et non plus une galerie d'images qui est une sélection des éléments réalisés soit par un expert, soit par un automate, soit par une personne qui sollicite un avis à un expert)'.*

C'est vers cette approche que les travaux de L. Zuyderhoff se sont orientés. Pour réaliser des mégaimages, le travail a consisté à réaliser des captures ligne par ligne pour, par la suite, les fusionner en une seule et unique image afin de former une mégaimage. Cependant, des erreurs d'alignement[4] obligent à capturer les images en prévoyant une zone de recouvrement qui permettra par la suite de réaliser un processus de coregistration.

La coregistration

Cette étape particulièrement complexe consiste à *'aligner ou mettre en correspondance deux images, appelées images maîtres et images esclaves recouvertes entièrement ou partiellement par une même zone. Cette zone sous-entend donc que les images représentent entièrement ou partiellement une même information.'*[Zuy03]

A l'entrée de cette étape, nous disposons d'une série d'images possédant chacune des caractéristiques communes de recouvrement. Grâce à la coregistration, nous disposons des coordonnées de positionnement de chaque image capturée. En positionnant chacune de ces images aux coordonnées indiquées par la coregistration, on obtient une "megaimage". Cette technique permet d'éviter l'échantillonnage orienté proposant ainsi à l'utilisateur de visualiser une partie de la lame et non plus quelques images capturées ci ou là.

Le lecteur désireux d'en apprendre plus sur la technique de coregistration utilisée est invité à consulter l'ouvrage [Zuy03].

La figure 2.3 illustre ce qu'est une mégaimage.

Cependant, dans la conclusion de son ouvrage, L. Zuyderhoff identifie un problème important. Même si la solution de coregistration qu'il propose offre des résultats satisfaisants, cette solution se limite à des mégaimages de petites tailles. En effet, l'implémentation proposée oblige l'ordinateur à garder en mémoire centrale l'ensemble des images capturées pour les coregistrer.

C'est de cette constatation que G. Decocq a débuté son travail. Dans son ouvrage [Dec04], G. Decocq propose une solution pour la création de ce qu'il qualifie de "Gigaimages". Cette solution propose de partir des megaimages

[3] Relatif à la moëlle épinaire *Cfr. Petit Larousse Illustré Edition 2005*

[4] Explications dans [Zuy03]

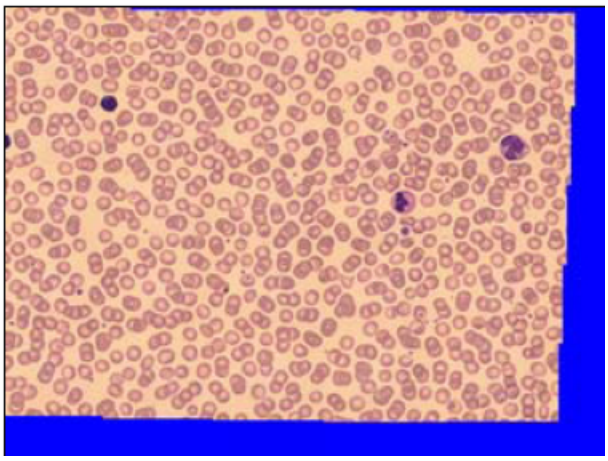

FIG. 2.3 – Exemple de mégaimage

réalisées à l'aide de la solution proposée par L. Zuyderhoff et de les coregistrer entre-elles en ne chargeant en mémoire centrale que les bords de celles-ci. Cette solution, séduisante en théorie ne s'est pas avérée efficace en pratique. En effet, deux types d'algorithmes de coregistration étaient appliqués pour la réalisation de ces "Gigaimages" ce qui d'une part causait un problème de temps de coregistration important et d'autre part, l'algorithme de coregistration utilisé par G. Decocq n'étant pas aussi performant que le premier, le résultat de la coregistration n'était pas toujours aussi bon qu'escompté.

La solution était donc de réutiliser l'algorithme de coregistration proposé dans [Zuy03] et de le transformer pour pouvoir réaliser une coregistration image par image. Cette solution fût donc réimplémentée plus tard par L. Zuyderhoff.

2.2.2 Le stockage

Les images ainsi coregistrées doivent être stockées sur un serveur. Le niveau de précision exigé pour ce type d'images entraîne une quantité d'informations très importante à stocker (plusieurs gigabytes pour une seule image). C'est pourquoi, il fallait trouver une solution permettant de compresser l'image tout en évitant une éventuelle perte de qualité.

Parmi les différents standards possibles, JPEG2000 était, de par ses caractéristiques et fonctionnalités[5] le plus à même à répondre aux exigences d'imagerie médicale. Au fur et à mesure des travaux, ce standard a été de mieux en mieux exploité. Au départ utilisé uniquement pour ses capacités à compresser sans perte de qualité, celui-ci fut utilisé par la suite pour compresser des images morceau par morceau. De plus, l'ensemble des fonctionna-

[5]Cfr. Matériel et méthode, Chap 3.

lités offertes par JPEG2000 en font un outil de compression très intéressant pour une application de télémicroscopie. Le standard JPEG2000 sera décrit dans la partie "Matériel et Méthode" de ce même ouvrage.

2.2.3 La visualisation

Dans son ouvrage, L. Zuyderhoff propose une application de visualisation permettant d'accéder à distance à des mégaimages réalisées tout en imitant du mieux possible les fonctionnalités d'un microscope conventionnel. L'application bien que très intéressante du point de vue des fonctionnalités, est assez limitée. En effet, deux problèmes majeurs sont à identifier.

D'une part l'application souffre d'un problème assez important de gestion de la mémoire. L'application part du principe que "ce qui a été décodé ne doit plus être décodé une deuxième fois". Le client garde en effet toujours en mémoire ce que le serveur lui a envoyé. L'occupation de la mémoire croît ainsi de manière linéaire jusqu'à arriver à saturation.

D'autre part, l'architecture initiale n'ayant pas fait l'objet d'une étude approfondie, la navigation présente certaines lenteurs qui rendent la réalisation d'un diagnostic quasi impossible.

Ce viewer était donc un prototype qui permit de donner quelques pistes pour la réalisation de l'application de ce présent travail.

2.3 Résumé des différents travaux

La figure 2.4 résume de manière schématique les différents travaux ayant participé au projet de télémicroscopie pour le laboratoire de cytologie de Mont-Godinne.

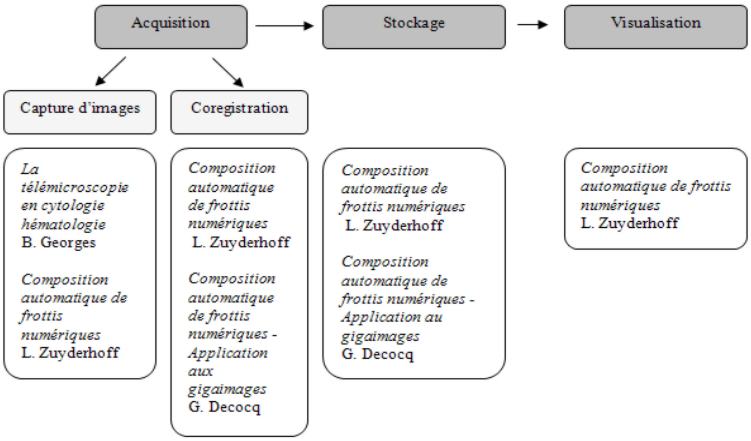

Fig. 2.4 – Travaux relatifs au projet de télémicroscopie de Mont-Godinne

2.4 Conclusion

Depuis plusieurs années, quelques travaux se sont succédé dans la réalisation d'une application de télémicroscopie pour le laboratoire de cytologie de Mont-Godinne. Dans l'état actuel des choses, nous pouvons considérer que nous disposons de mégaimages[6] compressées et stockées grâce au standard JPEG2000 sur un serveur dédié. C'est pourquoi, dans la continuité de la réalisation de ce travail, l'objectif suivant sera la création d'une application client/serveur exploitant les possibilités de JPEG2000 et permettant la visualisation à distance de ces mégaimages.

La partie "Analyses et résultats" de cet ouvrage présente une solution à ce type d'application.

[6]Dans [Dec04] l'auteur avait qualifié ces images de Gigaimages. Afin de rester constant dans les termes employés dans ce travail, nous conserverons l'appellation "megaimage".

Deuxième partie

Matériel et méthode

Chapitre 3

La compression d'images, le standard JPEG 2000

3.1 Introduction

L'amélioration constante des technologies repousse de plus en plus les limites de l'informatique. Depuis quelques années, nous assistons à une progression fulgurante des capacités des médias de stockage (Disques durs, DVD, Mémoires flash) et des vitesses des réseaux de communication. A l'heure actuelle, il n'est pas rare de rencontrer des ordinateurs domestiques équipés de disques durs atteignant des capacités de plusieurs centaines de gigabytes et connectés à un réseau type ADSL atteignant des vitesses de transfert très importantes.

Dans le même temps, des techniques de compression de données de plus en plus sophistiquées sont apparues, permettant à un utilisateur de limiter l'espace qu'occupent celles-ci et par conséquent limiter les temps de transfert réseau.

De par ce paradoxe, on est en droit de se demander pourquoi la compression des données occupe une place aussi importante dans un contexte où, les réseaux de communication fonctionnent à des vitesses impressionnantes et où les médias de stockage atteignent des capacités titanesques.

Dans ce chapitre, nous allons tenter de répondre à cette question en illustrant les besoins réels que représente la compression des données. Nous aborderons par la suite la présentation d'un nouveau format de compression d'images, à savoir JPEG2000 et étudierons son impact dans le monde médical. Finalement, nous terminerons par une petite conclusion résumant la pensée générale de ce chapitre.

3.2 La compression des données : une nécessité

La compression des données s'applique à de nombreux domaines comme la musique, l'imagerie, la vidéo. Dans la suite de ce travail, nous nous intéresserons tout particulièrement à la compression des images numériques.

Les images numériques, peuvent devenir très volumineuses lorsque celles-ci tentent de représenter avec le plus de précision possible des captures du monde réel. Dans certains domaines tels que la médecine, les images numériques deviennent très utiles. Les médecins, biologistes, radiologues ont besoin d'images de plus en plus grandes, l'on parlera alors de mega-images : ces images dont les dimensions atteignent des proportions gigantesques. Ce constat étant fait, nous allons tenter de démontrer pourquoi la compression d'images a un attrait réel.

Pour [Gra00], la compression d'images est importante au niveau de deux facteurs : **l'occupation d'espace sur un média de stockage et le temps de transmission sur un réseau**.

Une image présentée sur un écran d'ordinateur est constituée d'un certain nombre de points, les pixels. Pour illustrer les explications nous partirons d'une photo classique sur ordinateur[1]. Cette photo de plus ou moins 1750 par 1200 pixels soit 2.100.000 pixels pour toute l'image, peut être codée de plusieurs manières différentes : noir et blanc, 256 couleurs, milliers de couleurs (65536 couleurs) ou en millions de couleurs.

Le tableau ci-dessous illustre la taille qu'occupe cette image encodée dans ces différentes manières.

Nbr de couleurs	**Nbr de bits par pixel**	**Volume**
Noir et blanc	1bit(allumé ou éteint)	256Kbytes
256 couleurs	8bits	2Mégabytes
65536 couleurs	16bits	4Mégabytes
Millions de couleurs	24bits	6Mégabytes

Tab. 3.1 – Encodage d'une image de 2.100.000 pixels

Ces chiffres sont éloquents. Sans compression, le stockage d'un simple album photo de famille occuperait un espace considérable sur un média de stockage. Disposer d'une technique de compression d'images qui n'altère pas la qualité de celles-ci tout en diminuant l'espace d'occupation est donc d'un réel intérêt.

L'explication du deuxième facteur découle directement de l'intérêt porté au premier : plus la taille d'un fichier est réduite, moins de quantité d'informations il y a à transmettre sur le réseau et par conséquent plus vite est

[1] De la même taille et de la même qualité qu'une photo traditionnelle chez le photographe 10cm x 15cm

transféré le fichier.

Nous remarquons donc que **la compression d'images représente un aspect fondamental dans le domaine de l'image numérique.** Dans cet optique, différentes techniques de compression existent. Celles-ci permettent **de réduire significativement la taille des fichiers** offrant tantôt une compression sans perte de qualité, tantôt une compression avec perte voire même parfois les deux.

Dans le point suivant, nous allons nous attarder sur l'étude d'un nouveau standard de compression très intéressant offrant de multiples fonctionnalités, à savoir la compression JPEG 2000.

3.3 Le standard JPEG 2000

Lorsque l'on évoque le sujet de la compression d'images, la première norme qui vient directement à l'esprit est la compression JPEG. Cette norme, créée il y a plus de dix ans par le Joint Photographic Experts Groups (JPEG) a déjà fait ses preuves dans bon nombre d'applications utilisant les images. Ses qualités de compression à des taux faibles et moyens s'avèrent encore aujourd'hui très compétitives. Cependant, à des taux plus élevés, les images ainsi compressées se révèlent être d'une piètre qualité. De plus, son manque de flexibilité et de fonctionnalités témoigne de son incapacité à satisfaire toutes les exigences des applications d'aujourd'hui.

C'est à partir de ce constat que les membres du groupe JPEG se sont remis à plancher sur l'élaboration d'une nouvelle norme de codage d'images plus flexible et performante : JPEG2000.

'Thus, the aim of the JPEG2000 working group is to develop a new image coding standard for different types of still images (bi-level, grayscale, color, multicomponent, hypercomponent), with different characteristics (natural, scientific, remote sensing, text rendered graphics, compound, etc.), allowing different imaging models (client/server, realtime transmission, image library archival, limited buffer and bandwith ressources, etc.) preferably within a unified and integrated system. This coding system is intended for low bit-rate applications and will exhibit rate-distortion and subjective image quality performance superior to existing standards' [CS99]

Nous allons donc tenter de donner une définition claire de ce qu'est le JPEG 2000 et présenter concrètement la norme. S'en suivra un exposé du panel des fonctionnalités proposées par ce nouveau standard. Une partie plus technique fera suite et présentera l'algorithme de compression et de décompression proprement dit.

3.3.1 Definition

JPEG 2000 (Joint Photographic Exprets Group 2000) est une nouvelle norme ISO/IEC de compression d'images avec ou sans perte de qualité créée par le groupe du même nom. Ce standard permet des gains de compression importants. En effet, à qualité d'image égale, on obtient un fichier étant 20 à 40% moins volumineux qu'un fichier JPEG classique.

JPEG 2000 offre également un grand nombre de fonctionnalités qui lui permettent de se distinguer nettement des autres techniques de compression d'images.

Notons également que '*JPEG2000 n'est pas une amélioration de JPEG, il constitue une autre manière d'analyser, de décomposer l'image pour la compresser avec ou sans perte en apportant des fonctionnalités complètement nouvelles, en particulier pour les transmissions à faible bande passante*' [Ren03]. En effet, JPEG2000 emploie le DWT (Discrete Wavelet Transform), une compression par "wavelet" (ondelettes) ce qui lui prévaut une meilleure qualité de compression que son prédécesseur JPEG qui utilise le DCT (Discrete Cosinus Transform).

JPEG 2000 s'affiche comme étant la solution concrète à tous les domaines touchant de près ou de loin à l'image numérique comme l'atteste la figure 3.1 issue de [Lur].

FIG. 3.1 – Les domaines

3.3.2 La norme

Débutée en 1997, la norme JPEG 2000 est encore loin d'être terminée. Bon nombre des douze parties qui la constituent sont encore à l'état de chantier. D'autres quant à elles, sont déjà des standards internationaux.

L'une des principales caractéristiques de cette norme est que ses créateurs ont choisi de ne définir que l'algorithme de décodage ainsi que le format des données compressées appelé généralement 'codestream' ou 'bitstream'. Cette approche laisse aux utilisateurs une assez grande liberté dans le choix du système de codage et ouvre la porte à la compétition pour l'élaboration

de codeurs optimaux, pour autant que ceux-ci produisent des codestreams conformes aux spécifications de la norme.

La norme est divisée en 12 parties dont certaines sont publiées et d'autres non. L'explication des différentes parties de la norme est tirée de [Ren03]

La **première partie** est publiée par l'ISO, elle définit **l'algorithme de décodage**, le **format de** ce que l'on appelle **le codestream** et le **format JP2** qui permet d'encapsuler le codestream.

La **deuxième partie** définit les **extensions** de la partie 1. C'est ici qu'on définit par exemple toutes les possibilités dans le domaine des metadonnées (Format JPX).

La **troisième partie** s'appelle **Motion JPEG 2000.** Elle définit un format de codage de séquences audiovisuelles (MJ2) en tant que succession d'images compressées par un système respectant la partie 1 de la norme. Cette partie a été publiée, c'est une norme ISO.

La **quatrième partie** définit les règles de **compatibilité avec la norme** JPEG 2000. Elle est maintenant publiée avec une série de fichiers tests.

La **cinquième partie** est constituée de **logiciels de référence**. Il existe en particulier deux implémentations de JPEG 2000, l'une en C (projet Jasper), l'autre en Java (projet JJ2000).

La **sixième partie** définit un **format de fichier pour les images dites "compound"** c'est-à-dire composées à la fois de textes, d'images, graphiques,... (JPM)

La **septième partie** aurait dû définir un support minimal de compatibilité pour les appareils à puissance limitée. Elle a apparemment été **abandonnée** en cours de route et ne sera pas publiée.

La **huitième partie (JPSEC)**, consacrée aux problèmes de sécurité, a été lancée en décembre 2001 à Sidney.

La **neuvième partie (JPIP)**, consacrée aux outils d'interactivité, aux API et aux protocoles, a été lancée en mars 2003 à Séoul.

La **dixième partie (JP3D)**, a été lancée en décembre 2001 à Sidney. Cette partie parle du traitement des images volumétriques.

La **onzième partie (JPWL)**, a été lancée en décembre 2001 à Sidney. Cette partie traite du transfert des images dans les réseaux Wireless.

La **douzième partie (ISO Base Media File Format)** qui est en fait une modification de la troisième partie est en cours d'élaboration.

La **treizième partie** est la standardisation pour la recommandation d'un encodeur de base appelé 'royaltee free'.

A l'heure actuelle, les premières parties (1 à 7) de la norme sont clôturées et permettent déjà d'utiliser JPEG2000 dans des applications. Les autres parties (à partie de la 8), sont des extensions de la norme et sont toujours à l'état de recherche.

3.3.3 Les principales fonctionnalités

Dans sa partie 1, la norme définit une multitude de fonctionnalités très intéressantes. En voici une liste non exhaustive. Certaines fonctionnalités sont illustrées par des exemples dont quelques-uns d'entre eux sont des comparatifs mettant en évidence les apports par rapport à JPEG.

Une première fonctionnalité est sans conteste **la qualité des images JPEG 2000 à des taux de compression élevés.** En effet, JPEG 2000 utilise une nouvelle technique de compression, le DWT ou 'Transformation par Ondelettes Discrètes'. Cette nouvelle technique de compression fera l'objet d'une explication dans la partie traitant de l'algorithme d'encodage et de décodage. Ce nouveau procédé lui prévaut un pas de plus sur son petit frère JPEG qui lui, utilise une technique permettant de donner une approximation de la valeur des pixels ce qui ne donne de bons résultats que lorsqu'il est utilisé à des taux de compression faibles voire moyens. La figure 3.2 illustre clairement ces propos. En effet, à taux de compression équivalent, on voit nettement que JPEG2000 offre des résultats nettement supérieurs JPEG.

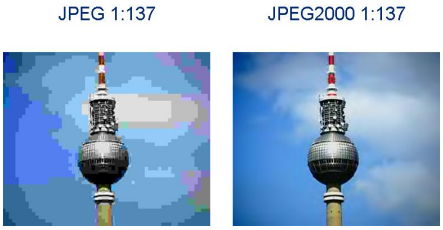

FIG. 3.2 – Comparaison JPEG et JPEG 2000 à même taux de compression. Image tirée de [MI]

Une deuxième fonctionnalité est la capacité, pour le même encodeur, de compresser **avec ou sans perte de qualité**[2]. Dans certains domaines tels que la médecine, l'altération de la qualité de l'image peut faire disparaître certaines informations pertinentes. Dans d'autres par contre, là où une légère dégradation de l'image peut être tolérée, la compression avec perte de qualité permettra à l'utilisateur de réaliser des gains d'espace importants.

Une troisième fonctionnalité permet d'encoder une image selon des **régions d'intérêt**[3]. Certaines parties d'une image sont plus importantes que d'autres. Celles-ci pourront être définies comme région d'intérêt c'est-à-dire reproduites avec moins voire sans perte de qualité. Il est ainsi possible d'atteindre des taux de compression très élevés tout en gardant une très

[2]Lossy (avec perte), Lossless (sans perte)
[3]ROI : Region of Interest

bonne précision dans les régions les plus importantes aux yeux de l'utilisateur [DS01].

La figure 3.3 montre l'image d'un papillon dont le bout de l'aile droite et la tête ont été sélectionnés comme étant des régions d'intérêt.

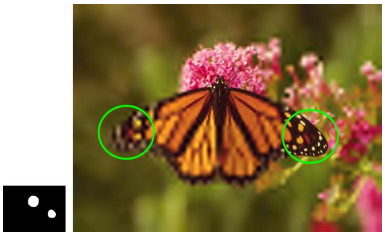

FIG. 3.3 – Le décodage ROI. Image tirée de [MI]

Une quatrième fonctionnalité, est **la protection contre les erreurs**. JPEG 2000 se veut très robuste face aux éventuelles erreurs pouvant apparaître dans le codestream suite à un transfert réseau par exemple.

La figure 3.4 illustre l'efficacité de JPEG2000 face aux éventuelles erreurs dans le codestream. Cette expérience tirée de [MI] a pour but de compresser une même image avec JPEG et JPEG2000 insérant volontairement dans chacun des codes obtenus une erreur : 16 Bytes ont été mis à 0 au milieu des deux fichiers. JPEG est incapable de faire face au problème et d'afficher correctement l'image. JPEG 2000 quant à lui, ne montre aucun problème face à l'erreur.

FIG. 3.4 – Gestion des erreurs. Image tirée de [MI]

Une cinquième fonctionnalité est **l'accès aléatoire rapide au codestream**. Celle-ci permet à l'utilisateur de ne décoder que les parties d'image qui l'intéressent. Cette fonctionnalité peut s'avérer très utile lorsque l'on manipule de grandes images et que l'on ne désire décoder qu'une partie de celles-ci.

Une sixième fonctionnalité, est **l'ordonnancement du codestream ou bitstream**. L'explication de cette fonctionnalité est tirée de [DD03]. 'La norme propose différentes manières d'organiser le bitstream d'une image. Il est possible de commencer le bitstream par une version de basse qualité, ou de faible résolution de l'image, l'affinant ensuite en précision. On peut également placer les composantes les unes à la suite des autres. On peut enfin exiger un ordonnancement spatial, qui commence par placer toute l'information concernant une zone de l'image avant de passer à une autre.

La norme offre également un autre avantage très important : le passage d'un mode de progression à un autre ne nécessite pas la décompression de l'image mais seulement un réordonnancement des paquets de données.'

Une septième fonctionnalité, sans doute l'une des plus importantes de la norme est le **décodage progressif**. Cette fonctionnalité est liée à la cinquième ou à la sixième fonctionnalité et offre la possibilité à l'utilisateur de décoder son image petit à petit suivant un certain ordre.

Quatre types de décodage peuvent être envisagés à partir d'un même codestream (par composante, par couche de qualité, par résolution, par progression spatiale). La figure 3.5 illustre cette fonctionnalité.

Une huitième fonctionnalité permet d'ajouter des **metadata**[4] aux fichiers JPEG2000. Cette fonctionnalité peut s'avérer utile lorsque l'on veut ajouter des commentaires à une image par exemple.

Une neuvième fonctionnalité est la possibilité pour JPEG 2000 **de traiter un grande nombre d'images** . La norme permet de travailler avec des images en couleurs, en niveaux de gris mais également des images composées de zones différentes (ex : texte + image).

Bon nombre d'autres fonctionnalités existent. Nous en avons dressé une liste des plus importantes. Cette liste montre très clairement l'avancée de JPEG 2000 par rapport aux autres normes de compression.

3.3.4 L'algorithme

Les points précédents nous ont permis de donner une définition claire de JPEG 2000 et d'en énumérer les principales fonctionnalités. Il est maintenant intéressant d'aller plus loin dans la démarche d'explications en essayant de décrire le fonctionnement de l'algorithme de JPEG 2000.

L'explication de l'algorithme d'encodage s'inspire de [DD03] et de [Gra00].

Comme nous l'avons déjà évoqué précédemment, la norme de JPEG 2000 ne définit que l'algorithme de décodage de JPEG2000 laissant ainsi libre les

[4]Version anglaise de métadonnée

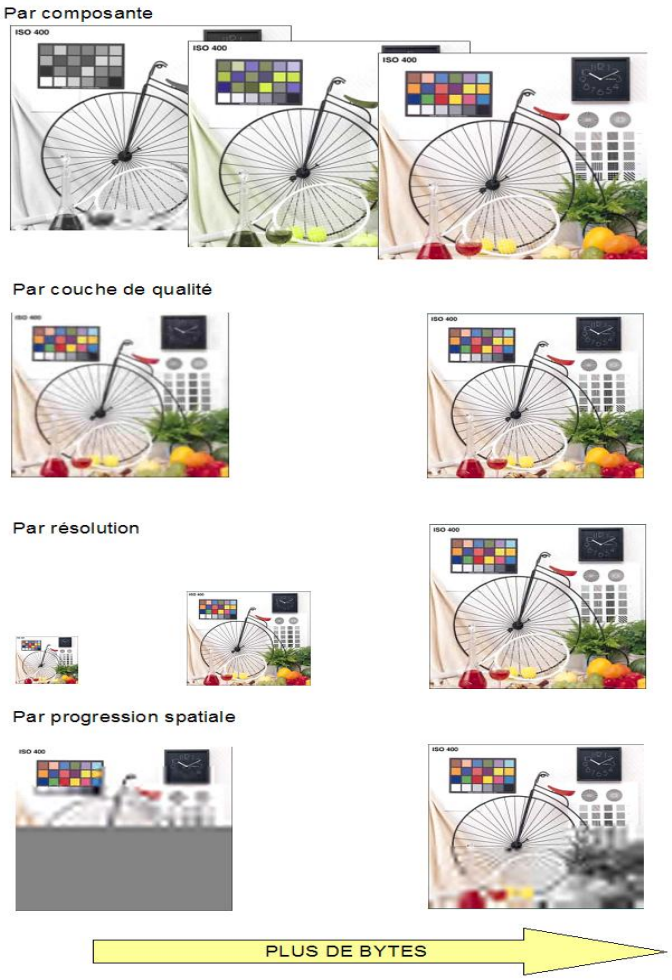

FIG. 3.5 – Le décodage progressif. Images tirées de [Tau03]

possibilités d'implémentation de l'encodeur pour autant que le codestream ainsi généré soit conforme à la norme. Cependant, dans un soucis de clarté d'explication, il est plus simple d'expliquer l'algorithme dans son processus d'encodage. C'est pourquoi nous présentons un algorithme de codage typique compatible avec JPEG2000 Partie 1.

Schéma de l'encodeur

Le schéma de l'algorithme JPEG 2000, est en fait une succession d'algorithmes pouvant se décomposer de la manière illustrée dans la figure 3.6.

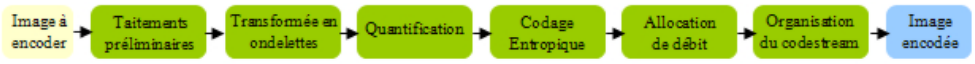

FIG. 3.6 – Schéma de l'encodeur JPEG2000

Traitements préliminaires

Le traitement préliminaire a pour objectif de 'préparer' l'image à être compressée. Cette phase est elle-même divisée en trois parties (Figure 3.7).

FIG. 3.7 – Traitements préliminaires

L'image à encoder peut, si celle-ci est très grande, occuper plus de place mémoire que la quantité disponible pour l'encodeur. Pour pallier ce problème, JPEG2000 propose une étape appelée **le tiling.** L'image principale est alors découpée en plusieurs petits rectangles de même taille[5] ne se recouvrant pas appelés tiles ou tuiles. Par la suite, ces tiles seront compressées indépendamment les unes des autres utilisant pour chacune d'entre elles ses propres paramètres de compression, comme s'il s'agissait d'images n'ayant aucun rapport. Celles-ci seront pour finir réunies dans un même codestream. La figure 3.8 illustre cette phase de tiling.

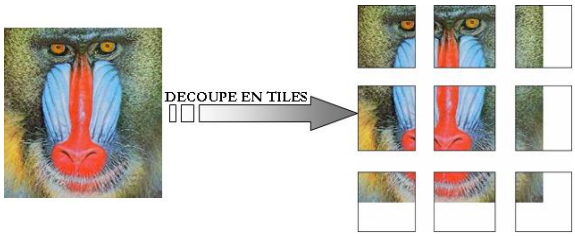

FIG. 3.8 – Le tiling

Encoder une image JPEG2000 non divisée en tiles revient à encoder une seule tile : l'image elle-même. C'est pourquoi dans la suite de cet exposé, nous traiterons le cas de l'image non divisée en tiles. Le fait de posséder plusieurs tiles n'est jamais qu'un bouclage sur l'algorithme de compression pour chaque tile.

[5]Sauf pour les images situées sur le bord droit et le bord inférieur

Afin de mieux comprendre les deux étapes suivantes, attardons-nous à la représentation d'une image en composantes. Une image est le résultat de la superposition d'une ou plusieurs composantes, jusque 2^{14} (Figure 3.9). Chaque composante consiste en une matrice de coefficients. Un coefficient étant la valeur de la luminosité associée à une composante d'un pixel de l'image. Les valeurs que peuvent prendre chaque coefficient sont des Integer signés ou non.

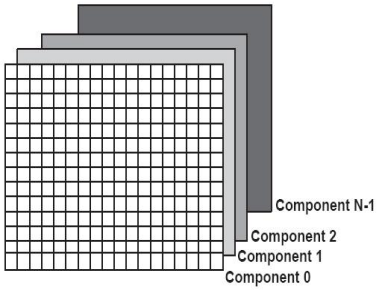

FIG. 3.9 – Le modèle d'image en composantes

Le modèle le plus couramment utilisé est le RGB. L'image ainsi représentée est constituée de trois composantes : le rouge (RED), le vert (GREEN) et le bleu (BLUE). La figure 3.10 représente la décomposition d'un pixel dans les trois composantes.

FIG. 3.10 – Décomposition d'un pixel dans l'espace de composante RGB

D'autres modèles de composantes existent notamment le RGBA. Ce modèle ajoute en plus des trois composantes citées précédemment, une composante alpha apportant l'effet de transparence. Dans la suite de cet exposé, nous baserons les explications sur le modèle le modèle RGB qui est le plus couramment utilisé.

La décomposition de l'image en composantes étant posée, poursuivons par l'explication de la seconde étape. Comme nous l'avons expliqué précédemment, les coefficients des matrices de composantes sont des Integer signés ou non. Lorsque ceux-ci sont non signés, **l'offset level**, est utilisé afin de ramener ces coefficients en logique signée, c'est-à-dire centrés autour

de zéro. Ce décalage est opéré afin de simplifier certains traitements comme la gestion de l'overflow.

L'étape suivante, la **transformée de couleurs** propose deux types de transformation : l'une **avec perte** et l'autre **sans perte de qualité**. Cette transformée est optionnelle mais peut s'avérer intéressante afin de décoreller les composantes entre elles.

La première, avec perte de qualité, permet d'obtenir une représentation de l'image dans un espace luminance[6]/chrominance[7]/chrominance plus adapté à la compression des données. Dès lors, en JPEG2000, une transformation irréversible de couleurs (ICT[8]) est réalisée (Figure 3.11) : c'est la transformation de RGB vers Y C_b C_r. En effet, les composantes Y C_b C_r sont moins dépendantes statistiquement les unes des autres et permettent ainsi une compression plus efficace. Les données sont alors représentées en virgules flottantes ce qui implique déjà des pertes lors du codage.

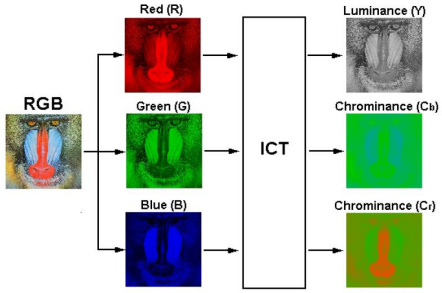

FIG. 3.11 – Transformation ICT d'une image

La deuxième sans perte de qualité, permet quant à elle d'approximer l'ICT afin d'éviter toute perte d'informations.

Transformée en ondelettes

JPEG2000 utilise le DWT (Discret Wavelet Transform). *'L'idée de base de la DWT consiste à séparer les basses et les hautes fréquences d'une image. Intuitivement, basses et hautes fréquences peuvent être comprises de la manière suivante. Les basses fréquences correspondent à une version grossière de l'image originale dans laquelle les valeurs des pixels ont été moyennées et où aucune variation brusque n'est observée d'un pixel à l'autre. Les hautes*

[6]Quotient de l'intensité lumineuse qu'émet une source par sa surface apparente. Elle s'exprime en nits.

[7]Terme technique définissant en fait les couleurs provenant du signal RVB. Cr et Cb : Coefficients de chrominance rouge et bleu.

[8]Irreversible Color Transform

fréquences quant à elles contiennent toute l'information sur les détails de l'image. On comprend aisément que plus d'informations sont contenues dans la version basses fréquences que dans celle ne fournissant que les détails de l'image. On dit que l'énergie de l'image est concentrée dans les basses fréquences.'[DD03]

'L'objectif de DWT est donc de concentrer l'information de l'image en une zone très localisée de manière à pouvoir ensuite compresser fortement les zones ne contenant que peu d'informations.' [DD03]

Ainsi, le DWT est appliqué en filtrant chaque ligne et chaque colonne de 'l'image préparée' avec un filtre de haute fréquence et de basse fréquence. Puisque le résultat donne un double échantillon de l'image, la sortie de chaque filtre est divisée en deux. La figure 3.12 illustre le principe de l'algorithme DWT.

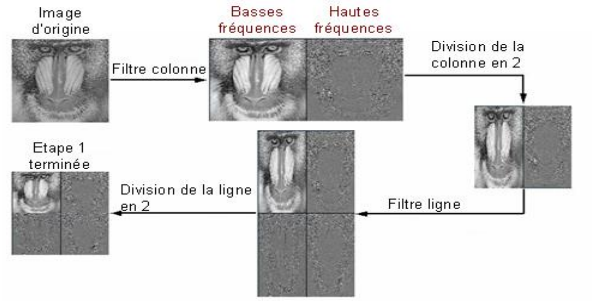

FIG. 3.12 – Discret Wavelet Transform (DWT)

Le nombre de fois qu'est appliqué l'algorithme dépend de l'implémentation utilisée pour la compression. La figure 3.13 montre le DWT appliqué deux fois de suite.

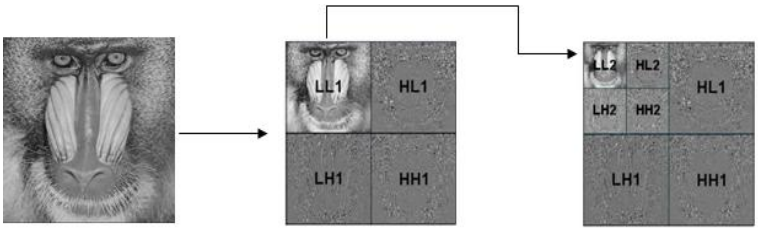

FIG. 3.13 – Application de la DWT deux fois de suite

C'est la DWT qui permet à JPEG2000 d'autoriser l'accès à l'image dans différentes résolutions. La résolution 0 étant la basse fréquence ligne/colonne du dernier filtrage DWT. Dans la figure 3.13, LL2 est la plus petite résolution que JPEG2000 puisse offrir.

Les filtres autorisés sont soit la paire (9,7) irréversible de Daubechies ou la paire (5,3) réversible de Le Gall. La première définit un filtre passe-bas à 9 coefficients et un filtre passe-haut à 7 coefficients ; tous deux à coefficients irrationnels. D'un autre côté, la paire (5,3) de filtres passe-haut et passe-bas est à coefficients rationnels. En règle générale, la première permet des taux de compression plus élevés pour une qualité donnée, mais seule la seconde est utilisable pour compresser sans perte.

La quantification

Lors du codage avec pertes, la précision sur les coefficients d'ondelettes est réduite par quantification scalaire uniforme. La fonction de quantification est présentée à la figure 3.14 où Δ est le pas de quantification de la sous bande (chaque sous bande peut avoir un pas de quantification différent), w est le coefficient d'ondelettes à quantifier et Q(w) l'index de quantification obtenu.

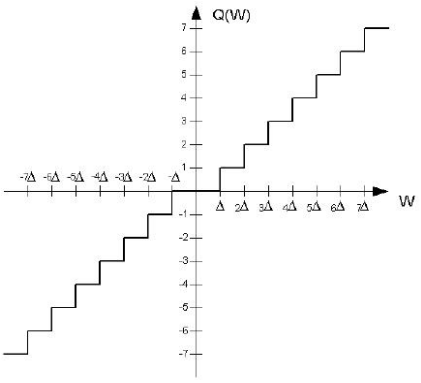

FIG. 3.14 – La fonction de quantification

Dans le cas d'un codage sans perte, le pas de quantification est égal à 1, ce qui signifie qu'aucune quantification n'est appliquée.

Le codage entropique

Lorsque les coefficients issus de la transformée en ondelettes ont été quantifiés, chaque sous bande est découpée en entités rectangulaires appelées code-blocks typiquement de taille 64x64 ou 32x32. Chaque code-blocks est encodé indépendamment au moyen d'un codeur entropique adaptatif au contexte. Brièvement, *'un codeur parvient à compresser une séquence de 0 et de 1 en réduisant la redondance présente dans cette séquence. Le fait qu'il soit entropique signifie qu'il réalise cette réduction de redondance en utilisant les probabilités d'occurrence de chaque symbole. Le fait qu'il soit adaptatif*

signifie que ces probabilités d'occurrence sont adaptées dynamiquement au cours du processus de codage. Enfin, le fait qu'il soit "avec contexte" signifie que la probabilité d'un symbole dépend de la valeur des symboles voisins.' [DD03]. Notons que le terme "probabilité" est à prendre au sens "fréquence".

L'allocation de débit

L'algorithme d'allocation de débit a pour but la création de paquets de données tels qu'ils sont définis dans la norme. Ces paquets sont créés grâce à la séquence binaire issue du codage entropique. Le découpage à plusieurs endroits de cette séquence binaire permet de créer les paquets. Chaque paquet correspond à une certaine couche de qualité (le layer), d'un niveau de résolution d'une composante de l'image d'une même zone spatiale. Les zones spatiales de chaque niveau de résolution sont appelées precints. Afin de permettre un accès aléatoire rapide dans le codestream, le paquet est constitué d'un en-tête identifiant son contenu. Le reste du paquet contient les données compressées.

Organisation du codestream

A cette étape, nous disposons d'un ensemble de paquets. Ces paquets doivent être organisés d'une certaine manière afin de créer un codestream valide. L'ordre dans lequel vont être agencés les paquets va définir l'ordre de progression. C'est ici que la fonctionnalité 'décodage progressif' prend sa forme.

Le codestream peut être organisé de quatre manières différentes : par couche de qualité, par niveau de résolution, par position ou par composante.

Lorsque le choix de l'organisation du codestream a été réalisé, il suffit d'écrire celui-ci sur le disque. Notons tout de même que ce choix n'est pas figé à tout jamais. En effet, il est possible de changer l'ordre de décodage de l'image de deux manières : soit en changeant l'ordre des paquets pour créer un nouveau codestream soit en utilisant la fonctionnalité d'accès aléatoire au codestream permettant, grâce aux en-têtes des paquets, d'accéder aux paquets désirés.

3.3.5 Subdivision d'une image

La figure 3.15 illustre les différentes parties d'une image après la compression. Cette image possède quatre résolutions.

L'image de départ a été découpée en *tiles* (Figure 3.8). Ceux-ci sont prétraités afin d'optimiser la compression. Chaque tile est alors découpée *en sous bandes* par le DWT. Chaque sous-bande est ensuite découpée en *code blocks* et compressée par le codage entropique. L'allocation de débit regroupe ces code blocks en *paquets* et l'organisation du codestream a lieu afin de

définir l'ordre de progression. Les codeblocks sont regroupés en **precints** afin d'accéder plus facilement aux zones spatiales de l'image.

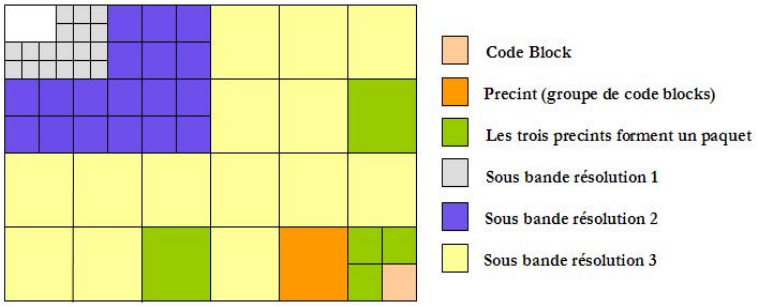

FIG. 3.15 – Structure d'une tile après compression

3.3.6 Structure du codestream

La norme dans sa partie 1 définit le format du codestream. La figure 3.16 inspirée de [Gra00] représente l'arbre de décomposition du codestream. Chacun des blocs est délimité par une série de marqueurs. Ces marqueurs ne seront pas cités ici. Cependant, le lecteur désireux de connaître les spécifications de ces marqueurs peut se rendre sur [COM00].

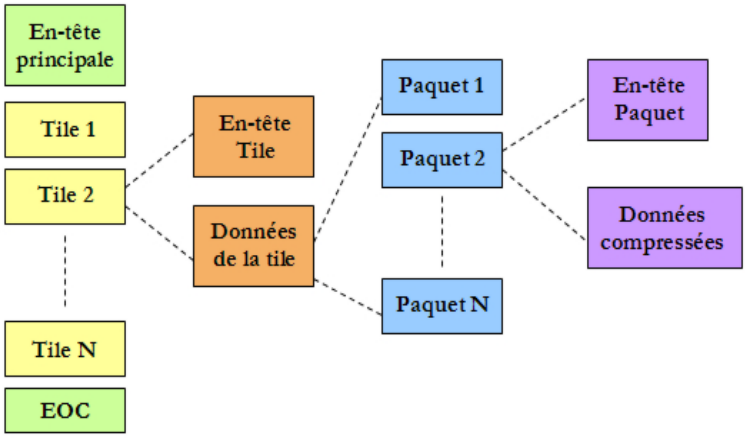

FIG. 3.16 – Arbre de structuration du codestream

3.3.7 Une compression - plusieurs décompressions

Une caractéristique très importante de JPEG 2000 est qu'il est possible à partir d'une même image compressée d'obtenir plusieurs décompressions. Il est en effet possible à partir d'un même codestream de décompresser l'image dans plusieurs qualités et même plusieurs résolutions. De plus, grâce à la fonctionnalité d'accès aléatoire, il est possible de ne décompresser que certaines parties de l'image. Ces fonctionnalités peuvent s'avérer très utiles surtout lorsque l'image compressée peut être visionnée par plusieurs appareils ayant des caractéristiques différentes. La figure 3.17 inspirée de [Gra00] illustre l'utilité de cette multi-décompression.

FIG. 3.17 – Multi-décompression d'un même codestream

La figure 3.18 tirée de [Tau03] illustre les codeblocks qu'il faut décompresser à partir du fichier JPEG2000 afin d'obtenir la partie d'image désirée dans la qualité et la résolution souhaitées.

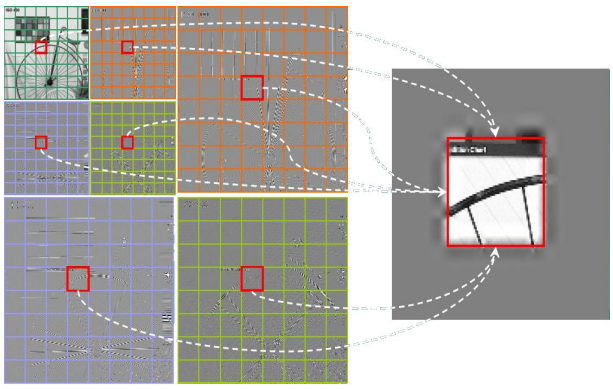

FIG. 3.18 – Codeblocks à décompresser

3.3.8 JPIP - Le protocole interactif de JPEG 2000

Jusqu'à présent, nous nous sommes intéressés à la flexibilité de JPEG2000 (accès aléatoire et ordonnancement du codestream, décodage progressif). Dans un monde où les réseaux sont omniprésents, il serait intéressant de disposer d'un protocole de communication exploitant cette flexibilité. A cet effet, la partie 9 de la norme JPEG2000 définit un outil de transfert d'images et de metadatas : **le protocole de communication JPIP**.

Dans ce point, nous allons brièvement expliquer le fonctionnement de JPIP. Le lecteur désireux d'approfondir le sujet peut se référer aux articles [Tau02] et [TP]

JPIP est un protocole de communication client/serveur pouvant être implémenté sur HTTP, mais conçu également pour être transporté avec d'autres protocoles. Ce protocole permet aux serveurs de répondre aux demandes des clients en renvoyant des fichiers JPEG2000 complets ou des morceaux d'images exploitant ainsi au maximum les propriétés offertes par JPEG2000.

La figure 3.19 tirée de [AWA] illustre une architecture utilisant JPIP. Dans cette architecture, le client envoie au serveur via une syntaxe bien définie, des requêtes spécifiant les parties d'image ou les metadatas dont il a besoin. Le serveur quant à lui, sélectionne le morceau de fichier JPEG2000 compressé correspondant à la requête du client. Ce morceau est ensuite envoyé au client qui cache ce fichier sur le disque, le décompresse et l'affiche. La cache disque permet ainsi d'éviter de demander à nouveau les mêmes morceaux d'image au serveur.

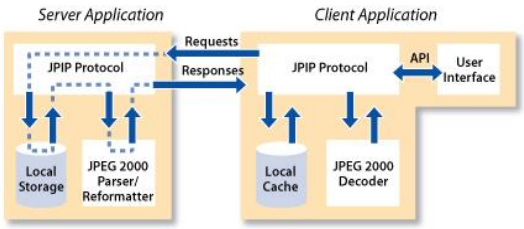

FIG. 3.19 – Architecture JPIP

En terme réseau, la possibilité d'accéder à certaines zones de l'image permet de réaliser des gains de bande passante et de vitesse de transfert importants. Le client quant à lui verra son volume d'informations et ses temps de calculs réduits. Ceci est particulièrement intéressant lorsque l'on travaille avec des images très grandes nécessitant un espace de stockage que le client ne possède pas toujours et un temps de décompression important si l'on veut décompresser toute l'image. En effet, il n'est pas toujours nécessaire d'envoyer l'intégralité de l'image au client dans sa meilleure résolution et sa meilleure qualité. Le cytologiste par exemple, pourra consulter une version de

l'image en petite résolution et sélectionnera par la suite les morceaux d'image qu'il désire visualiser en résolution plus grande et dans ce cas, seulement cette partie de l'image est downloadée.

Notons simplement que JPIP permet également l'utilisation du mode sans état (stateless) et du mode avec état (statefull), permettant de réaliser une gestion de la cache sophistiquée pour éliminer la transmission redondante de données.

La partie 9 de la norme JPEG2000 définit également une manière d'**indexer** les codestreams JPEG2000. Ces index permettent d'accéder aléatoirement aux paquets désirés. Cette manière de procéder permet de réaliser son propre protocole d'accès aux images là où toutes les fonctionnalités offertes par JPIP ne sont pas nécessaires. Le parsing n'est alors plus effectué en temps réel à chaque requête du client. Le fichier est 'préparsé[9]'.

3.3.9 Les implémentations JPEG2000

A ce jour, plusieurs implémentations de JPEG2000 existent. Nous allons énumérer les plus importantes.

- **JJ2000** est une implémentation de JPEG2000 réalisée en Java par Canon, l'EPFL et Ericsson. Cette implémentation est fournie avec le code source Java permettant ainsi à un programmeur de l'utiliser à des fins non commerciales. JJ2000 a été sélectionné comme l'un des deux produits standards de la partie 5 de la norme JPEG2000. `http://jj2000.epfl.ch`.

- **Jasper** est une implémentation de JPEG2000 en C réalisée par Image Power, Inc. et l'University of British Columbia. Cette implémentation open-source permet aux programmeurs de la distribuer, la modifier et l'utiliser à des fins non commerciales. Jasper, est le deuxième produit standard sélectionné dans la partie 5 de la norme. `http://www.ece.uvic.ca/~mdadams/jasper/`.

- **Kakadu** est une implémentation de JPEG2000 en C++ réalisée par l'un des fondateurs de la norme JPEG2000, David Taubman. Kakadu a été implémenté avec les principales fonctionnalités de JPIP ce qui permet une navigation client/serveur dans les images JPEG2000. `http://www.kakadusoftware.com/`.

- **OPENJpeg librairie** est un codec JPEG2000 open-source écrit en langage C par le laboratoire TELE de l'Université Catholique de Louvain (UCL). Ce codec a été développé dans le but de promouvoir l'utilisation de JPEG2000. OpenJpeg est sous licence BSD ce qui permet

[9] A la création du fichier JPEG2000 un exécutable est lancé permettant d'accompagner le fichier JPEG2000 d'un fichier d'index

à un programmeur d'utiliser et de modifier la librairie même pour des applications commerciales. `http://www.openjpeg.org`.

Le site `http://jj2000.epfl.ch/jj_links/index.html` propose une liste d'autres solutions possibles (plugins, codecs, solutions hardwares).

3.4 JPEG2000 et le monde médical

L'ensemble des caractéristiques de JPEG 2000 en fait un standard très utile dans de nombreux domaines et tout particulièrement l'imagerie médicale. Les médecins, biologistes, radiologues sont amenés à manipuler des images de plus en plus grandes, difficiles à stocker et à transmettre sur un réseau de communication.

De plus, l'imagerie médicale exige un niveau de qualité très élevé, requiérant ainsi une compression sans perte de qualité. En effet, la dégradation de l'image et l'apparition d'artefacts pourraient conduire à des erreurs de diagnostic inacceptables dans un domaine où la vie des patients est en jeu.

A cet effet, JPEG 2000 est capable de **compresser des images sans perte de qualité et ce, avec de bonnes capacités de compression**. JPEG2000 possède également des fonctionnalités très intéressantes directements applicables à l'imagerie médicale. Voici une liste de quelques applications de fonctionnalités dans le domaine médical :

- L'utilisation de JPIP permet de n'avoir qu'une seule copie des images sur un serveur centralisé. Ces images étant de grandes tailles, les clients peuvent accéder aux parties de l'image qui les intéressent permettant ainsi d'avoir des stations de visualisation très fluides en réduisant les temps de calcul et la quantité d'informations à stocker. De plus, JPIP permet de réaliser des applications de type 'télémédecine' permettant par exemple à un expert de consulter une image à distance afin de rendre un diagnostic.

- La fonctionnalité **région d'intérêt** offre déjà des ouvertures très intéressantes dans le domaine. Les images étant de très grandes tailles, même compressées, celles-ci risquent d'occuper une place importante sur le disque surtout si l'utilisateur désire du "sans perte". Grâce aux régions d'intérêt, on pourra compresser sans perte les parties intéressantes, les autres parties quant à elles moins intéressantes pourront être compressées normalement ou avec perte.
  De plus, les régions d'intérêt pourront être décompressées avant les autres permettant ainsi à l'observateur de gagner un temps précieux en ayant déjà les parties de l'image qui l'intéressent. Le résultat du diagnostic n'en sera que plus rapide.

– JPEG2000 offre la possibilité d'ajouter des **metadatas** au codestream. Ceci ouvre la voie à la possibilité d'ajouter des annotations sur des zones de l'image et même des commentaires sur toute l'image entière. Certaines applications, comme celles liées aux standards DICOM (Digital Imaging and Communication In Medicine) en charge de la création et de la maintenance d'un standard facilitant l'échange de données médicales, ont leurs propres méthodes sophistiquées pour traiter ces métadatas. Le comité JPEG travaille avec le comité DICOM pour s'assurer que ces deux standards importants pourront facilement être intégrés.

– Le codestream JPEG 2000 peut être **ordonné** pour transmettre des images de résolution inférieure, ou de qualité réduite, bien avant que l'image entière ne soit transmise. Ceci aide fortement les applications de navigation, et signifie qu'un seul fichier est nécessaire pour diverses applications.

– De **nombreuses formes d'images** peuvent être compressées avec JPEG 2000 : images radiologiques, cytologiques et d'autres types d'images médicales.

Il est à noter également que l'imagerie médicale nécessite des outils de navigation performants dans les mega-images[10]. Ces outils pourront tirer parti de l'ensemble des fonctionnalités proposées par JPEG2000 (accès aléatoire au codestream, réorganisation du codestream sans recompression, décodage progressif). Cependant, l'utilisation de JPEG2000 par les médecins, biologistes, radiologues doit leur permettre de réaliser un diagnostic plus rapide ou tout aussi rapide qu'avec les moyens traditionnels. Dans le cas contraire, l'utilisation de JPEG2000 n'est pas très intéressante. La partie analyse et résultats de ce même travail propose une solution à la navigation dans les mega-images tirant profit des fonctionnalités de JPEG2000.

3.5 Conclusion

Dans ce chapitre nous avons démontré la nécessite de disposer d'une méthode de compression d'images. Dans cette optique nous nous sommes tout particulièrement attardés sur le standard JPEG2000. En effet, les caractéristiques de JPEG2000 en font un standard très prometteur dans de nombreux domaines notamment le domaine médical. Il est maintenant possible de réaliser avec JPEG2000 des applications qu'il aurait été impossible de réaliser auparavant par exemple [Zuy03] et [Dec04]. L'ensemble des fonctionnalités

[10]Terme généralement donné aux images de très grandes tailles

offertes, ainsi que sa grande flexibilité, ouvrent la porte à la réalisation d'applications très élaborées comme l'application réalisée dans le cadre de ce travail et décrite dans la partie 'Analyse et résultats'.

Chapitre 4

Optimisation des performances : le caching et le prefetching

4.1 Introduction

Il y a encore quelques années, alors que les débits des réseaux étaient limités à des vitesses relativement lentes, la réalisation d'applications client-serveur exigeantes en temps de réponse[1] était quasi inconcevable. Depuis quelques années, l'augmentation des débits dû à l'apparition de technologies type ADSL et l'amélioration des réseaux locaux, furent des facteurs stimulants pour la réalisation de ce type d'applications.

Dans le chapitre précédent, nous avions noté que la compression des données permettait d'augmenter sensiblement la vitesse de transfert réseau par la réduction des données à transférer. Dans certaines situations cependant, la compression des données ne peut satisfaire à elle seule l'exigence d'un temps de réponse très rapide. Il peut en effet arriver que, dans certaines applications, les temps de réponse deviennent trop longs pour diverses raisons (serveur surchargé, redondance d'informations, réseau lent) rendant l'application quasi inutilisable. Pour essayer de contrer ces problèmes, des techniques permettant d'optimiser les transferts client-serveur sont apparues.

Dans ce chapitre nous allons tenter d'exposer des techniques permettant d'optimiser sensiblement les performances d'applications réseau au temps de réponse critique. Dans cette optique, nous tenterons de définir et d'expliciter les principes du **caching des données** et d'en donner les principaux avantages. Nous évoquerons par la suite plusieurs méthodes possibles pour la gestion d'une cache. Nous présenterons également la technique du **préfetching**[2]. Nous tirerons enfin quelques conclusions.

[1] Le temps entre l'envoi de la requête et la réception complète des données

[2] Terme anglais que l´on pourrait traduire par ´pré-traitement/chargement´

4.2 Le caching

En informatique, la notion de caching peut s'appliquer à beaucoup de domaines. Dans cet exposé, nous limiterons nos explications au caching client-serveur. D'autres modèles de caching comme par exemple celui du processeur, ne seront pas abordés ici. Les notions de base sont cependant identiques.

Dans certains types d'applications, un client est parfois amené à consulter plusieurs fois la même donnée à intervalles réguliers. Les gains alors réalisés grâce à la compression pourraient très vite se perdre au détriment d'une redondance de transfert de la donnée. Ces mêmes demandes répétitives couplées à un nombre croissant de clients engendrent un accroissement important du trafic réseau et de la charge de travail au niveau du serveur pouvant se traduire par un temps de réponse beaucoup trop long pour certains types d'applications.

4.2.1 Principe et définition

Afin d'illustrer le principe du caching, partons de la métaphore suivante : un chercheur réalise un travail nécessitant des documents se trouvant en bibliothèque. Chaque fois qu'il désire consulter une page d'un livre, il peut envoyer un mail à la bibliothèque en demandant de la photocopier et de la lui envoyer par courrier. Dès la réception, il la consulte puis, la jete immédiatement à la poubelle. Quelques heures plus tard, il se rend compte qu'il désire à nouveau consulter cette page. Celle-ci ayant été jetée, notre chercheur est obligé d'envoyer une nouvelle demande à la bibliothèque. Afin d'éviter un double travail pour la bibliothèque et une perte de temps pour le chercheur, il eut été préférable que celui-ci dispose d'une farde permettant de stocker ces documents et de les retrouver rapidement.

Transposé dans le monde informatique, le fait de classer les documents reçus dans une farde à proximité du client s'apparente à ce que l'on appelle **du caching.**

Selon Microsoft : *"Caching is a technique widely used in computing to increase performance by keeping frequently accessed or expensive data in memory. In the context of a Web application, caching is used to retain pages or data across HTTP requests and reuse them without the expense of recreating them."*

De manière simple, le caching a pour but de **stocker des données fréquemment ou récemment utilisées.** Ce stockage peut être réalisé soit sur le disque dur soit en mémoire vive. Lorsque le client a besoin d'une donnée, la cache est d'abord compulsée. Si celle-ci possède les données nécessaires, aucune requête n'est envoyée au serveur. Dans le cas contraire, le serveur est interrogé. Nous pouvons donc constater que le temps de réponse pour une donnée qui se trouve déjà en cache est inférieur au temps de réponse si

la même donnée devait être demandée au serveur.

La figure 4.1 illustre le fonctionnement du caching à l'aide d'une machine à état. Le client souhaite consulter la donnée A.

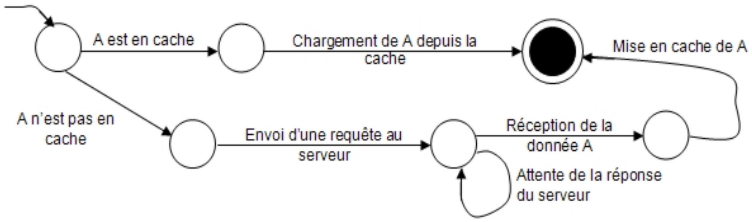

FIG. 4.1 – Le client souhaite consulter la donnée A

Dans la métaphore du chercheur, arrive un moment où la farde de celui-ci est remplie. Il faut alors décider d'un document à éjecter. A cet effet, plusieurs politiques existent, nous les décrirons un peu plus tard. Avant toute chose, afin de bien cerner le caching, nous allons, dans le point suivant en évoquer les principaux avantages.

4.2.2 Avantages du caching

Les avantages du caching sont assez évidents. Pour des applications dans lesquelles plusieurs consultations d'une même donnée risquent de se présenter, le caching des données présente des avantages non négligeables au niveau de trois facteurs :

Le premier facteur est **la réduction de la charge de travail du serveur**. A petite échelle, lorsqu'un seul client est connecté au serveur, le gain est peu significatif. Cependant, si l'on étend à des dizaines de clients connectés au même serveur, la diminution du nombre de requêtes envoyées par la disponibilité de l'information en cache pour le client, permet de diminuer sensiblement le travail du serveur.

Le deuxième facteur se situe au niveau du **désengorgement de la bande passante**. En effet, avec le caching, beaucoup moins d'informations circulent sur le réseau. Sur un réseau lent ou un réseau utilisé par beaucoup d'utilisateurs, l'impact du caching peut se révéler très intéressant.

Le troisième facteur est **la diminution du temps de réponse pour le client**. Cet avantage est soit la conséquence des deux autres c'est-à-dire, plus la charge du serveur est réduite et plus la bande passante est libre, plus les temps de réponse sont rapides, soit la conséquence d'une disponibilité immédiate de l'information désirée en cache.

4.2.3 Politique de gestion de la cache

Lorsqu'une application client-serveur utilise le modèle de caching, le client a la possibilité de limiter la taille de sa cache. Lorsque celle-ci est remplie, le client doit décider quel bloc de données il va éjecter. A cet effet, plusieurs politiques de gestion de la cache existent.

Une première méthode, est la méthode aléatoire. Cette méthode est la plus simple. Elle consiste à éjecter au hasard un bloc de données de la cache.

Une deuxième méthode, est la LRU (Least Recently Used). Le principe consiste à marquer par un timestamp[3], un bloc présent dans la cache lorsqu'il est utilisé. Lorsque la cache est pleine, le bloc le moins récemment utilisé sera alors éjecté pour faire place au suivant.

Une troisième méthode est la LFU (Least Frequently Used). Le principe consiste à incrémenter un compteur associé à chaque bloc de données en cache lorsque ce bloc est utilisé. Lorsque la cache est pleine, le bloc le moins fréquemment utilisé est alors éjecté pour faire place au suivant.

Une quatrième méthode est le FIFO (First In First Out). Le principe de cette méthode consiste à éjecter le bloc le plus ancien dans la cache.

Une cinquième méthode est l'utilisation d'un TTL (Time To Live). Le principe consiste à associer à chaque bloc de données un compteur de temps de vie spécifiant le temps durant lequel le bloc de données doit rester en cache. Lorsque le temps de vie du bloc est à expiration, ce bloc est éjecté de la cache.

Notons simplement que dans une application, certaines politiques de gestion sont plus efficaces que d'autres.

4.2.4 Le préfetching

Principe

Comme nous l'avons noté, le client bénéficie plus rapidement de la donnée désirée lorsque celle-ci se trouve en cache. Dans la figure 4.1, on voit que lorsque le client ne dispose pas de la donnée nécessaire dans sa cache, celui-ci passe par une phase d'attente de réponse du serveur puis de téléchargement. Afin d'éviter au maximum cette procédure qui risque d'être trop longue (réseau ou serveur lent), il serait judicieux de cumuler le système de cache à un autre système permettant de remplir celle-ci à l'avance. C'est le principe du **prefetching**.

Grâce à des modèles mathématiques de prefetching spécifiques à l'application, il peut être possible, de profiter **des moments d'inactivité du client** pour télécharger certaines données que l'algorithme de prefetching aurait sélectionnées suivant des critères qui lui sont propres. L'algorithme

[3]Indication du temps auquel se produit l'évènement

de prefetching réalise donc deux choses : **créer un liste des données à télécharger et télécharger ces données.**

Plusieurs critères peuvent être sélectionnés afin de décider quelles sont les informations à télécharger à l'avance. Pour donner un exemple simple en faisant le lien avec le chapitre traitant de JPEG2000, il était possible de sélectionner dans une image, des régions d'intérêt. L'algorithme de prefetching pour une application de navigation dans l'image pourrait décider de télécharger ces régions d'intérêt à l'avance pour les mettre en cache permettant ainsi une visualiation plus fluide de l'image.

Notons simplement que les algorithmes de prefetching peuvent très vite devenir lourds pour le client lorsque ceux-ci réalisent des calculs très importants pour déterminer les blocs de données à télécharger. C'est pourquoi, un algorithme de préfetching ne doit être implémenté que dans des applications pour lesquelles les temps de réponse sont critiques.

Complétons à présent la figure 4.1 en y ajoutant les deux états du client à savoir l'état 1 : actif et l'état 2 : inactif. Pendant l'état d'activité, l'algorithme traditionnel du caching sera utilisé tandis que lorsque le client est en état d'inactivité[4], l'algorithme du préfetching sera lancé. La figure 4.2 présente l'ajout de ces différentes caractéristiques. Le prefetching client ne doit pas perturber l'utilisateur lorsqu'il visualise des données. A cet effet, l'algorithme de prefetching doit pouvoir être interrompu à tout moment si le client désire retourner du mode inactif au mode actif.

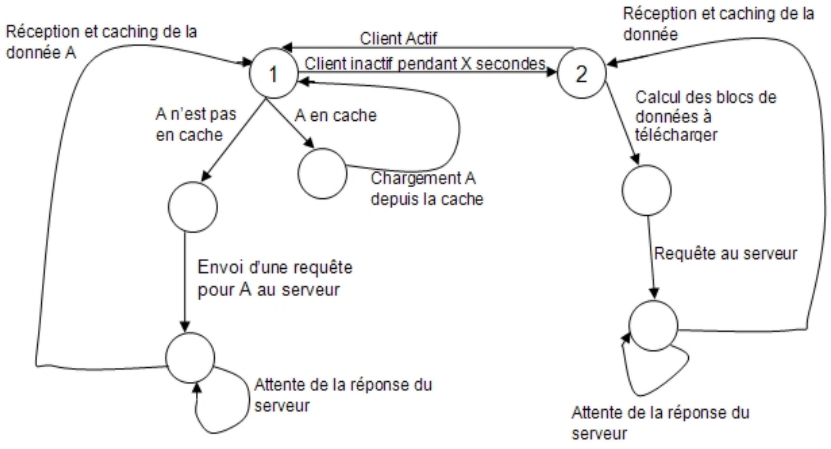

FIG. 4.2 – Machine à état avec caching et prefetching

[4]Décidée lorsque le client n'a plus réalisé d'actions pendant X secondes

Avantages

Comme nous l'avons déjà cité auparavant, le prefetching présente un avantage lorsque les temps de réponse pour l'application client sont critiques. Le prefetching va donc 'combler' deux principaux problèmes de lenteur liés au modèle client-serveur.

Le premier problème que le prefetching va tenter de résoudre se situe au niveau de **la vitesse de transmission du réseau**. Lorsque celle-ci est très lente, l'application client peut devenir inutilisable si l'on devait télécharger les blocs de données au fur et à mesure de leur nécessité.

Le deuxième problème que le prefetching va tenter de résoudre se situe au niveau du **temps de réponse du serveur**. Le serveur peut en effet ne pas être très rapide soit de par ses caractéristiques techniques soit parce qu'il est solicité par un grand nombre de clients.

4.3 Conclusion

Dans ce chapitre, nous avons présenté différentes techniques permettant à des applications client-serveur au temps de réponse critique de fonctionner le plus efficacement possible.

Dans cette optique, nous avons mis en avant les avantages que représente l'emploi du caching des données dans une application client-serveur. Cette technique peut en effet s'avérer très efficace pour une application dans laquelle plusieurs consultations d'une même donnée risquent de se présenter. Les gains peuvent alors se remarquer à plusieurs niveaux : la diminution de la charge du travail du serveur, le désengorgement de la bande passante et surtout un meilleur temps de réponse pour le client.

Afin d'améliorer encore l'aspect 'temps de réponse', nous avons présenté une amélioration possible du caching appelé le prefetching. En effet, grâce à des modèle mathématiques spécifiques, il est possible de déterminer avec une probabilité plus ou moins certaine les blocs de données que le client sera amené à consulter dans un avenir plus ou moins proche et ainsi de profiter des moments d'inactivité du client pour les télécharger.

Lorsqu'une application client-serveur est développée, il peut être judicieux d'étudier la possibilité d'intégrer ces différentes techniques surtout s'il s'agit d'une application avec un temps de réponse critique. Il ne faut cependant pas oublier qu'il s'agit de techniques ajoutant un surplus de travail au client. C'est pourquoi, une analyse judicieuse doit être réalisée afin d'éviter un alourdissement pour l'application client.

Dans la partie analyse et résultats de ce mémoire, nous étudierons la mise en place d'un système de caching couplé à un système de prefetching sophistiqué pour une application de navigation dans des mega-images.

Troisième partie

Analyses et résultats

Chapitre 5

Objectifs et choix réalisés

5.1 Introduction

Tout au long du deuxième chapitre de l'état de l'art, nous avons pu nous rendre compte de l'état d'avancement du système de télémicroscopie du laboratoire de Mont-Godinne suite aux différents travaux réalisés par d'anciens mémorants. Nous en avons conclu que le module principal à développer était celui de la visualisation.

Suite à différentes discussions avec les responsables du laboratoire de cytologie[1], des objectifs ont été définis afin de cadrer le développement de l'application. Dans ce chapitre, nous allons dresser un bilan des principaux objectifs qui ont guidé la réalisation de ce travail. Suite à cela, nous poserons et justifierons les principaux choix qui permettront de les atteindre le plus efficacement possible.

5.2 Présentation des objectifs

La mise en évidence des principaux objectifs n'est pas chose aisée. Il faut à la fois tenir compte des caractéristiques des individus susceptibles d'utiliser le logiciel mais également des limites potentielles imposées par certaines contraintes techniques.

L'objectif principal de l'application de visualisation est de permettre à un spécialiste extérieur au laboratoire, de visualiser des mégaimages représentant des lames virtuelles précédemment capturées et stockées sur un serveur d'images. Cette personne habituée à diagnostiquer des lames porte-objets à l'aide d'un microscope conventionnel, devra être à même, grâce à l'outil de visualisation proposé, de poser un diagnostic le plus juste possible afin d'aider le biologiste du laboratoire dans son examen. C'est pourquoi, l'application actuelle doit reposer sur des principes permettant un diagnostic de **qualité, rapide et efficace**. Les objectifs décrits ci-dessous ont dès lors été

[1] Mr Chatelain B., Mr Cornet Y.

principalement guidés par les exigences des biologistes, les fonctionnalités de base du microscope conventionnel et en s'inspirant d'exemples d'applications telles que celles décrites dans l'état de l'art de cet ouvrage et ce, afin que l'application soit utilisable dans un contexte médical.

Les principaux objectifs sont :

- Un accès à distance et sécurisé
- La fluidité de la navigation
- Une navigation intuitive
- La prise en compte de la diversité des environnements
- Les outils supplémentaires

5.2.1 Un accès à distance et sécurisé

Cet objectif est un aspect qui va influencer de manière très importante la conception globale de l'application. Dans cette perspective, le paradigme client-serveur est un moyen très puissant pour faire communiquer des applications dans un environnement distribué[2]. Grâce à un protocole de communication bien précis, le client et le serveur peuvent ainsi dialoguer afin de s'échanger des informations. A cet effet, les seuls paramètres nécessaires à la connexion sont l'adresse IP du serveur ainsi que le numéro de port identifiant l'application sur ce même serveur.

Il est également important de noter que le serveur devra être à même de permettre à plusieurs clients de se connecter simultanément pour visualiser des images.

Le caractère confidentiel des données transférées par une application de type médical exige un niveau de sécurité assez élevé. C'est pourquoi, il faut s'assurer que seules les personnes autorisées peuvent avoir accès aux mégaimages présentes sur le serveur. Un système d'accès sécurisé par login et mot de passe est donc une exigence minimale.

5.2.2 La fluidité de la navigation

L'application de visualisation se doit d'imiter du mieux possible les caractéristiques d'un microscope conventionnel. C'est pourquoi, la navigation[3] dans l'image doit être la plus fluide[4] possible. Cet objectif est sans conteste le plus complexe à réaliser car les images à visualiser font plusieurs centaines voire plusieurs milliers de mégabytes ce qui dans un environnement

[2]Environnement dont les éléments sont géographiquement distants mais reliés par réseau

[3]Par navigation nous entendons les déplacements suivant l'axe X-Y et les changements d'objectifs

[4]La fluidité signifie que nous voulons éviter au maximum les temps d'attente d'affichage de l'image pour le client

client-serveur demande beaucoup de ressources et ce, à plusieurs niveaux à savoir **la rapidité du réseau, les performances du serveur et du client.** Tous ces paramètres doivent êtres optimisés au maximum tant au niveau hardware que par des techniques software telles que le caching[5] et les fonctionnalités avancées offertes par le standard JPEG2000[6] comme par exemple le décodage progressif.

Bien que ces techniques soient performantes, il faut néanmoins rester réaliste quant à la performance du matériel hardware. Il serait, en effet, utopique d'envisager une fluidité de la navigation avec des postes clients de type 486 connectés à un réseau modem 56Ko. C'est pourquoi, l'objectif de fluidité ne pourra être atteint que dans des conditions minimales de performance hardware à savoir une connexion ADSL, un poste client cadencé à 1Ghz et 256Mo de RAM et un serveur d'image performant.

5.2.3 Une navigation intuitive

Comme nous l'avons déjà précisé, l'application sera utilisée par des biologistes habitués à réaliser leur diagnostic à l'aide d'un microscope conventionnel. C'est pourquoi, l'application réalisée devra imiter du mieux possible les caractéristiques offertes par ce type d'appareil à savoir le déplacement suivant l'axe X-Y et les changements d'objectifs.

Il faut également noter qu'avec ce type d'application, le biologiste ne dispose pas du support physique (la lame porte-objets) lui permettant de connaître sa position dans l'échantillon analysé. C'est pourquoi, une fonctionnalité permettant de visualiser l'échantillon dans son ensemble sera ajoutée. C'est ce que nous avons qualifié de slide view[7]. De plus, en rendant ce slide view cliquable, l'application permettra au médecin de se déplacer directement à une position choisie de l'échantillon.

5.2.4 Tenir compte de la diversité des environnements

L'application cliente est destinée à être installée dans des environnements différents. Celle-ci est en effet susceptible d'être utilisée dans différents hôpitaux et laboratoires. Cette diversité des environnements engendre des configurations pouvant varier énormément au niveau des résolutions d'écran utilisées (800x600, 1024x768, 1400x1050) mais également au niveau du système d'exploitation présent sur le poste de travail (Windows, Linux, Mac,...). C'est pourquoi, le développement de l'application doit prendre en compte cette diversité en choisissant des outils adéquats permettant à l'application de s'adapter.

[5] Cfr. Matériel et méthode, Chapitre 4
[6] Cfr. Matériel et méthode, Chapitre 3
[7] Cfr. Etat de l'art, Chapitre 1

5.2.5 Les outils supplémentaires

Ces outils supplémentaires présents sur l'application cliente, aideront les biologistes à poser leur diagnostic. Il est cependant à noter que ces outils ne sont pas de grande utilité si les objectifs de base, c'est-à-dire ceux cités précédemment, ne sont pas atteints. Cependant, il est important d'en tenir compte dès à présent pour pouvoir dès le début de la conception de l'application, prévoir l'ajout de ces différents outils de manière aisée.

Ces outils pourront ainsi être définis ultérieurement par les biologistes au fur et à mesure de leur besoin. Nous pouvons à titre d'exemple en citer quelques-uns : capture d'images, compte globule, mesure de l'échantillon, travail sur les couleurs de l'image (RGB, luminosité, contraste).

5.3 Choix réalisés

Dans le point précédent, nous avons défini une série d'objectifs qui devront être respectés par l'application sous peine d'être difficilement exploitables dans le monde médical. Pour ce faire, certains choix techniques et conceptuels doivent être réalisés. Dans ce point, nous allons étudier les choix qui vont mener à la réalisation d'une application en phase avec les objectifs fixés. Nous tenterons également de motiver chacun de ces choix.

Avant toute chose, il est important de se remémorer que l'application à développer fait suite à une série de travaux déjà réalisés[8]. Le standard JPEG2000 nous est dès lors imposé. Notons malgré tout que ce choix aurait sans nul doute été celui de ce travail. JPEG2000 possède en effet des fonctionnalités intéressantes offrant un certain nombre d'avantages à une application manipulant des images médicales[9]. Les choix présentés ci-dessous partent donc du principe que JPEG2000 est un choix préalablement imposé.

5.3.1 Le langage de programmation JAVA

L'avantage principal offert par Java est sans conteste sa portabilité. Un programme écrit dans ce langage est compilé une première fois pour fournir du byte-code assurant à Java sa portabilité vers de très nombreux environnements[10]. Chaque environnement possède son propre interpréteur appelé la machine virtuelle. Celle-ci permet de traduire ce byte-code en code exécutable par l'environnement cible. Il n'est donc pas nécessaire de recompiler le code pour chaque environnement. Ce premier avantage répond directement

[8] Cfr. Etat de l'art, Chapitre 2

[9] Cfr. Matériel et méthode, Chapitre 3 : JPEG2000 et le monde médical

[10] Windows 3.1/95/98/NT4/2000/ME/XP/CE, MacOS, Solaris, Linux, AIX, OS/2, IRIX, UnixWare, HP/UX, Digital Unix, AmigaOS, BeOS, OpenVMS, FreeBSD, SunOS, RiscOS

à l'objectif exigeant de tenir compte de la diversité des environnements possibles pour le poste client.

Java s'accompagne également de nombreuses bibliothèques très complètes pour la conception d'interfaces graphiques ce qui s'avère particulièrement adapté à la conception d'une interface la plus soignée et la plus intuitive possible exigée pour le poste client.

De plus, de par son paradigme orienté objet, Java permet de rédiger un code proprement découpé en modules ce qui s'avèrera particulièrement efficace pour l'implémentation de l'architecture que nous présenterons dans le chapitre suivant.

Enfin, les images de base étant encodées sous le format JPEG2000, le choix de Java s'avère judicieux dans le sens où une librairie totalement open source écrite dans ce langage permet de manipuler les images JPEG2000 à partir de son propre code. C'est la librairie JJ2000 [Rap].

Il ne faut cependant pas oublier que dans le langage Java, la gestion de la mémoire se fait de manière automatique par un mécanisme appelé le garbage collector. Ce langage ne permet pas de travailler directement sur cette gestion de la mémoire. Des problèmes avaient déjà été rencontrés dans l'application viewer réalisée par L. Zuyderhoff dans [Zuy03]. L'architecture devra donc être conçue de manière à prendre en compte ce problème. Nous y reviendrons dans les détails de l'architecture au chapitre suivant.

5.3.2 Décompression et transfert par résolution de tile

L'un des objectifs de l'application est d'offrir une visualisation fluide. Les temps de réponse[11] doivent dès lors être les plus courts possibles. De plus, dans une optique de performance il faut essayer d'occuper le moins de mémoire possible pour ne pas reproduire les problèmes rencontrés dans le viewer du travail [Zuy03].

Afin de répondre à ces différentes exigences, le standard JPEG2000 s'avère très utile de par les caractéristiques et les fonctionnalités qu'il propose. En effet, grâce à ce standard, il est possible de ne décompresser qu'une partie de l'image. Cette approche offre un certain nombre d'avantages :

- L'application étant scindée entre un client et un serveur, les images doivent transiter par le réseau. En ne transférant que les parties de l'image nécessaires à la visualisation, on diminue l'engorgement et les temps de transfert réseau.
- Les ressources matérielles nécessaires à la décompression sont concentrées uniquement sur les parties que l'utilisateur désire consulter.
- L'occupation de la mémoire est également optimisée. La mémoire ne contient que les parties d'image nécessaires à la visualisation.

[11] Par temps de réponse on entend le temps entre l'envoi de la requête du client au serveur et l'affichage de l'image à l'écran

Les images étant encodées sous forme de tiles[12], c'est tout naturellement que les tiles seront choisies comme unité de transfert pour les morceaux d'images.

Une fonctionnalité importante de l'application est de pouvoir visualiser l'image suivant plusieurs niveaux de zoom. C'est ici que la fonctionnalité de JPEG2000 permettant de visualiser l'image suivant plusieurs résolutions nous sera utile. Il n'est donc pas nécessaire de transférer l'entièreté de la résolution d'une tile visualisée. Chaque tile sera donc transférée suivant le niveau de zoom et donc de résolution exigée par l'utilisateur. Les requêtes porteront ainsi sur le numéro d'une résolution d'un numéro de tile.

La notion de fluidité peut également être renforcée en décidant, lorsque l'utilisateur réalise un zoom avant ou arrière, d'utiliser des techniques permettant d'approximer l'image future avant que celle-ci ne soit rafraîchie. La technique d'interpolation de pixels sera alors utilisée. Les caractéristiques de cette technique sont présentées en annexe de ce travail. Précisons que pour le développement de l'application, la technique d'interpolation bilinéaire (présentée en annexe) a été choisie pour la qualité satisfaisante des images produites mais surtout pour sa rapidité de calcul.

5.3.3 Protocole propriétaire et générateur d'index

L'exigence pour l'application de fonctionner en mode client-serveur nous oblige à utiliser un protocole de communication entre les différentes entités. A cet effet, le standard JPEG2000 propose dans sa norme un protocole de communication exploitant l'ensemble des fonctionnalités offertes par ce standard. Ce protocole, appelé JPIP, s'avère cependant trop complexe à implémenter pour le type de requête dont nous avons besoin. Les seules requêtes porteront en effet sur le numéro de résolution d'un numéro de tile. L'utilisation d'un protocole propriétaire avec une syntaxe simple est donc amplement suffisante. Ce protocole fera l'objet d'une explication plus approfondie dans le chapitre suivant.

Afin de pouvoir accéder aux différentes parties du codestream, il est nécessaire de disposer d'un fichier d'index tel que la norme le prévoit. Ce fichier permet en autre de préciser à quel endroit du codestream se trouve une résolution d'une tile précise. A cet effet, le laboratoire TELE de l'Université Catholique de Louvain (UCL) propose un générateur d'index écrit dans le langage C permettant à partir d'un fichier JPEG2000 de générer le fichier d'index associé. La syntaxe du fichier d'index ainsi générée est présentée en annexe. Nous nous rendons bien compte que l'utilisation du langage C limite la portabilité du serveur. Cependant, dans un premier temps, il s'agit du moyen le plus rapide pour arriver à des résultats concrets dans les délais octroyés. Il sera toujours possible dans une phase ultérieure, de porter ce

[12]La dimension des tiles étant paramétrable pour l'encodage

code dans le langage JAVA.

5.3.4 Le client décompresse, le serveur distribue

L'un des objectifs précédemment cité impose que l'application soit développée suivant le paradigme client-serveur : le serveur possédant l'ensemble des images compressées et le client étant la station de visualisation se connectant à ce même serveur. De par cette exigence on peut se demander quelle entité aura pour fonctionnalité de réaliser la décompression afin d'obtenir l'affichage de l'image le plus rapidement possible.

Lorsque le client envoie une requête au serveur afin d'obtenir une résolution d'une tile de l'image, deux possibilités s'offrent à nous :

1. Le serveur analyse la requête et consulte le fichier d'index, sélectionne le morceau de codestream correspondant, le décompresse et le transfère au client pour qu'il n'ait plus qu'à l'afficher.
2. Le serveur analyse la requête et consulte le fichier d'index, sélectionne le morceau de codestream correspondant, l'envoie au client qui le décompresse et l'affiche.

A première vue, on pourrait penser que les deux solutions sont équivalentes. Cependant, en y regardant d'un peu plus près, on s'aperçoit que l'une des deux solutions offre des avantages non négligeables.

La première solution a pour avantage de profiter des capacités hardware offertes par le serveur. Celles-ci étant généralement assez élevées. Cependant, le serveur peut être amené à répondre à plusieurs client connectés en même temps ce qui pourrait causer des problèmes de temps de réponse dus à l'engorgement de calculs subi par la décompression simultanée de plusieurs requêtes. De plus, nous avions précédemment évoqué que l'un des avantages offerts par la compression était de réduire les temps de transfert réseau[13]. C'est pourquoi, en laissant le serveur s'occuper de la decompression, on risque de perdre une partie des avantages et d'augmenter les temps de réponse c'est-à-dire de ralentir l'affichage de l'image. De surcroît, si le client travaille avec un système de cache, celle-ci se verra très vite saturée si elle doit stocker des morceaux d'image décompressés.

On voit dès lors que la deuxième solution offre des avantages incontestables et répond à l'ensemble des critères de performance escomptés. Cependant, comme nous l'avons déjà signalé auparavant, nous insistons sur le fait que le client doit malgré tout posséder des performances hardware minimales pour permettre à la décompression d'être réalisée dans des temps relativement satisfaisants.

Laisser le client décompresser s'avère donc la meilleure solution pour répondre à l'objectif de fluidité.

[13]Cfr. Matériel et méthode, Chapitre 3

5.3.5 Le caching côté client

Dans les chapitres précédents, nous avons pu énumérer les avantages offerts par l'utilisation du caching dans une application de type client-serveur[14]. Ces avantages s'avèrent très intéressants pour accroître les performances d'une application telle que celle-ci. L'utilisation du caching permettra dès lors de renforcer l'objectif de fluidité de la navigation dans l'image.

L'application stockera en cache les résolutions de tiles déjà téléchargées pour éviter de devoir les télécharger une seconde fois si l'application en a encore besoin. Cependant, dans un souci de rapidité, la première version de cette cache ne prendra pas en compte de politique de caching. L'ajout d'une politique pour la gestion de la cache pourra être réalisée dans des versions futures. Nous prendrons donc comme hypothèse que dans sa première version, la taille offerte pour la cache sur le disque dur sera illimitée.

5.4 Conclusion

A travers ce chapitre, nous avons énuméré les principaux objectifs que l'application finale devra respecter afin d'être utilisable dans un contexte médical. A cet effet, nous avons posé et justifié plusieurs choix qui permettront, lors de la réalisation de l'application, de s'orienter du mieux possible vers la réalisation de ces objectifs.

Certains choix comme par exemple le créateur d'index préfabriqué et la taille de cache illimitée ont été posés afin de limiter le développement de l'application dans le temps. Chacune de ces limitations posées ne sera cependant pas un frein au développement futur de l'application et pourra dès lors faire l'objet d'un ajout dans les versions à venir.

C'est dans cette perspective que nous rédigerons le chapitre traitant de la discussion de ce travail. Nous y émettrons certaines critiques vis-à-vis du travail réalisé et nous proposerons des pistes pour l'optimisation de certains objectifs.

[14] Cfr. Matériel et méthode, Chapitre 4

Chapitre 6

L'application viewer

6.1 Introduction

Dans le chapitre précédent, nous avons posé une série de choix nous permettant de respecter du mieux possible les objectifs fixés. Cependant, la construction d'une application ne se limite pas seulement à une série de choix. En effet, une phase de construction d'architecture est nécessaire afin d'intégrer l'ensemble des choix réalisés dans une structure globale. Cette architecture pourra également, de par sa conception, permettre d'améliorer certaines performances.

La réalisation de l'architecture doit également tenir compte d'éventuelles évolutions propres à l'application. C'est pourquoi, une structuration claire divisée en différents modules et une certaine flexibilité de sa structure permettront, par la suite, de pouvoir étendre et optimiser l'application plus facilement.

Ce chapitre met l'accent sur l'architecture déployée pour la réalisation de l'application de visualisation. En tenant compte des objectifs et choix réalisés dans le chapitre précédent, nous avons construit une architecture robuste et découpée en différents modules. Nous présenterons tout d'abord l'aspect général de l'architecture puis, nous présenterons plus en détail l'architecture développée pour la partie serveur et la partie client. Finalement, à travers différentes captures d'écran, nous présenterons l'aspect final de l'application réalisée.

6.2 Structure générale de l'application

Dans le chapitre précédent, nous avons évoqué la nécessité pour l'application d'adopter le paradigme client-serveur : le serveur possédant l'ensemble des mégaimages à visualiser et le poste client étant un ordinateur quelconque appartenant à un expert désireux de consulter les mégaimages présentes sur ce même serveur. Pour communiquer, le client et le serveur utilisent un pro-

tocole propriétaire, ce protocole étant le langage commun permettant aux deux entités de se "comprendre".

La figure 6.1 illustre l'aspect général de l'application telle que souhaitée par le laboratoire de cytologie.

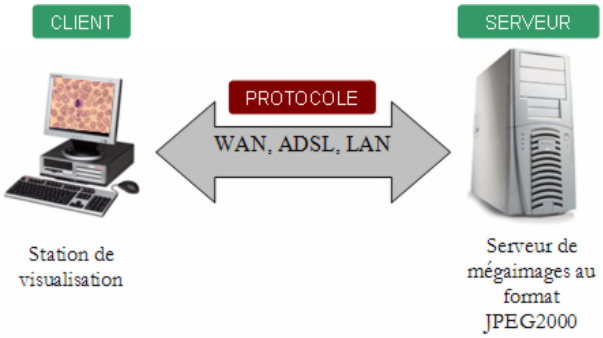

FIG. 6.1 – Vue générale de l'application

Dans les points suivants, nous allons décrire plus en détails l'achitecture développée pour le serveur et pour le client. Nous évoquerons brièvement le protocole développé.

6.3 Le serveur

La tâche principale du serveur est de gérer **l'envoi des parties d'images** réclamées par les clients à travers un réseau de communication grâce à **un protocole** bien défini. Dans cette optique, l'architecture du serveur devra être assez robuste pour pouvoir gérer **plusieurs clients émettant des requêtes simultanément**.

Comme nous l'avons déjà signalé, le caractère confidentiel relatif aux données médicales transférées par l'application nous oblige à instaurer du côté serveur, une certaine **limitation d'accès à ces mégaimages** par un système de login et de mot de passe permettant d'identifier chacune des personnes désirant se connecter.

La figure 6.2 illustre l'aspect général de l'architecture réalisée pour la partie serveur. On y distingue trois modules : **le connexion manager, le pool de thread et la gestion des requêtes**. Dans les points suivants, nous allons détailler chacun de ces modules afin d'en dégager les principaux concepts.

Reprécisons qu'à chacun des fichiers JPEG2000 est associé un fichier d'index permettant de connaître la structure précise du codestream JPEG2000. Ce fichier étant généré par un créateur d'index réalisé par le laboratoire TELE de l'Université Catholique de Louvain, nous n'entrerons pas dans

les détails de la réalisation de ce programme. Le lecteur désireux d'en apprendre plus sur les concepts liés à la réalisation de ce programme est invité à contacter les membres du personnel de ce laboratoire. Précisons simplement que lorsque le serveur est amené à ouvrir un fichier JPEG2000 sans fichier d'index associé, ce programme est invoqué et le fichier index est créé.

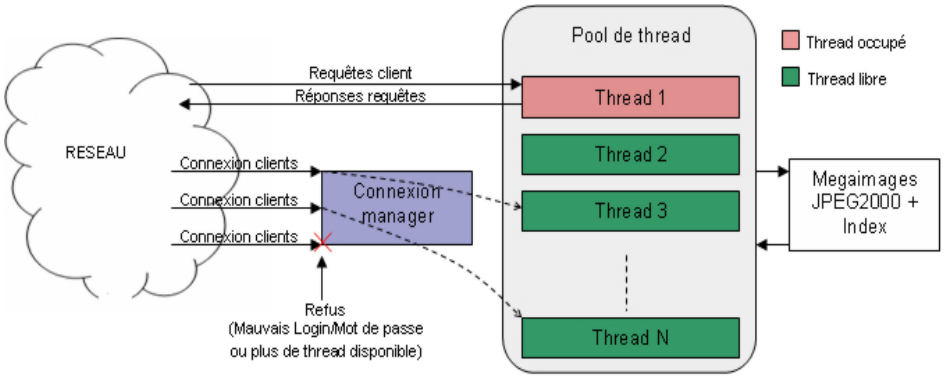

FIG. 6.2 – Architecture du serveur

6.3.1 Le connexion manager

Ce module peut être considéré comme la porte d'entrée du serveur. Le connexion manager est en effet le premier module contacté par le client lorsque celui-ci désire se connecter. A cet effet, le connexion manager joue deux rôles particuliers.

Le premier est **un rôle de contrôle d'accès**. Le connexion manager doit en effet vérifier que les paramètres login et mot de passe introduits par le client sont corrects avant de lui attribuer l'accès au serveur. Si ces données s'avèrent correctes, le connexion manager joue un deuxième **rôle de dispatcher** c'est-à-dire qu'il va vérifier si, dans le pool de thread, un thread est disponible afin de pouvoir gérer les requêtes du client. Si c'est le cas, un thread libre est attribué au client pour toute la durée de sa connexion. Dans le cas contraire, le client est averti et invité à se reconnecter plus tard.

6.3.2 Le pool de thread

Le pool de thread consiste en la création anticipée de plusieurs "Thread"[1] ensuite affectés aux requêtes émises par l'utilisateur. Le fait de disposer de plusieurs thread permet le développement d'applications nécessitant des

[1]Tâche Légère en français. processus léger, correspondant à l'exécution d'un petit programme, ou d'une routine d'un programme plus gros, indépendamment de celui-ci (on parle alors de multithread). Cfr. http ://www.tout-savoir.net

traitements en parallèle. Lorsque le serveur reçoit une requête de connexion de la part d'un client, un thread du pool lui est attaché par le connexion manager pour toute la durée du traitement. Lorsque cette connexion est relâchée, le thread rejoint à nouveau le pool.

L'utilisation d'un tel modèle permet une meilleure maîtrise des ressources allouées par le serveur et d'ajuster le comportement du serveur à l'activité des différents clients.

Notons également que le pool doit contenir autant de thread que de clients que l'on désire pouvoir gérer simultanément (Sur la figure 6.2, N clients peuvent être connectés simultanément).

Le cycle de vie d'un thread du serveur peut se résumer par le diagramme d'état illustré à la figure 6.3.

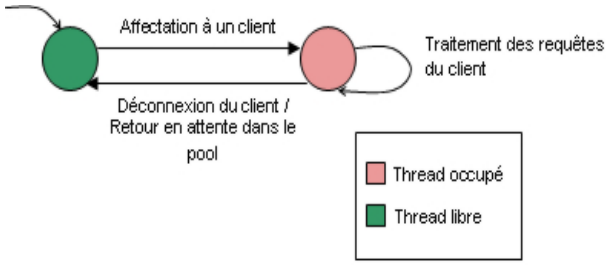

FIG. 6.3 – Diagramme d'état d'un thread du serveur

6.3.3 Le traitement des requêtes

Le serveur dispose d'un répertoire contenant l'ensemble des mégaimages disponibles. Comme nous l'avons déjà signalé, chacune de ces mégaimages est associée à un fichier d'index. Chacun des threads présents dans le pool de thread a accès aux mégaimages et aux fichiers d'index. Lorsqu'un thread du serveur est alloué à un client, celui-ci est amené à répondre à l'ensemble des requêtes de celui-ci. Ces requêtes exprimées sous un format particulier[2], portent sur des numéros de résolution de tiles.

Lorsque le serveur reçoit une telle requête, celle-ci est mise en correspondance avec une entrée dans le fichier d'index associé à la mégaimage consultée. Cette entrée permet au thread de savoir quelle partie du codestream répond à la requête du client. Ce morceau est alors envoyé. Comme nous l'avons déjà précisé dans les choix, le serveur ne s'occupe pas de la décompression des images.

La tâche attribuée au serveur ne demande pas d'utilisation de ressources très importantes. En effet celle-ci se compose de quelques entrées-sorties dans deux fichiers différents et d'un envoi par le réseau. C'est pourquoi, le

[2]Cfr. la syntaxe du protocole décrite plus loin

serveur peut gérer de manière optimale un ensemble assez conséquent de clients connectés simultanément.

6.4 Le client

Pour répondre à l'objectif de fluidité, la capacité du serveur à répondre aux requêtes dans les plus brefs délais est importante. Cependant, c'est au client que revient la lourde tâche de **décompresser** les parties d'image à visualiser. La décompression est en effet une tâche demandant un certain temps et utilisant beaucoup de ressources. De plus, il incombe également au client de gérer cette **visualisation** tout en tenant compte des **déplacements réalisés par l'utilisateur dans l'image**.

Dans ce point nous allons présenter l'architecture mise en place pour le développement de l'application client. Comme nous pourrons le constater, cette architecture présente des caractéristiques offrant des possibilités d'évolution de par sa découpe en modules. De plus, cette architecture permet d'améliorer les performances de fluidité de la navigation notamment en mettant la priorité sur les actions de l'utilisateur.

La figure 6.4 schématise l'architecture développée pour la réalisation de l'application client.

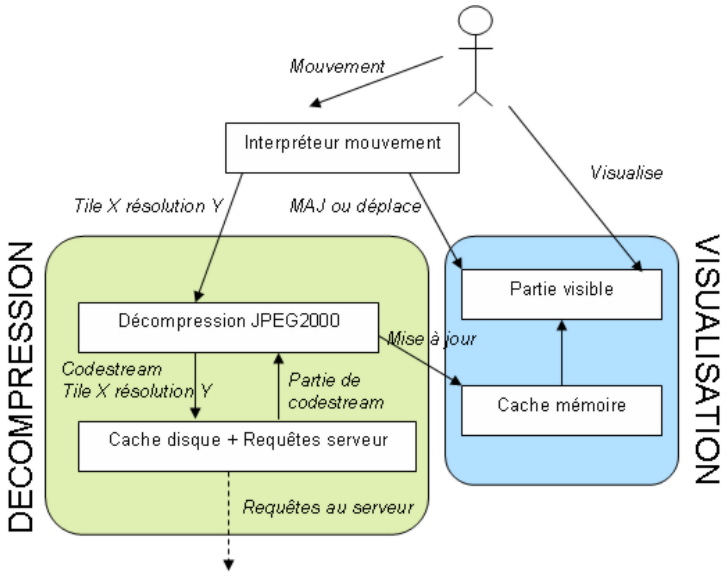

Fig. 6.4 – Architecture générale de l'application client

Sur ce schéma, il convient de distinguer deux parties dont l'objectif principal est différent : une partie gérant la décompression et une partie gérant la

visualisation. Ces deux parties étant liées entre elles par un troisième module appelé l'interpréteur de mouvements que nous décrirons plus tard.

La partie traitant la décompression se compose de 2 modules : un module gérant la décompression JPEG2000 et un autre traitant un système de cache disque ainsi que les téléchargements au serveur. Cette partie est donc chargée de décompresser les parties d'images nécessaires à la visualisation. L'originalité de cette décomposition vient du fait que le module cache disque rend des services au module de décompression. Afin que ces deux modules puissent travailler de manière indépendante, ceux-ci sont implémentés sous forme de thread.

Pour dialoguer entre eux, les modules utilisent un système de file de tâches à réaliser. Lorsque chaque module termine la tâche qu'il réalise, celui-ci consulte sa file d'attente pour vérifier si une tâche lui a été assignée. Si c'est le cas, il la traite sinon il attend qu'on lui attribue une tâche. Ce mode de traitement permet un travail asynchrone entre les modules. Notons également qu'afin de garder la priorité sur les déplacements de l'utilisateur, il est possible pour le module précédent de vider l'entièreté de la file qu'il a lui-même remplie pour le module suivant et ensuite remplir cette file avec les nouvelles tâches.

La partie gérant la visualisation. Cette partie permet de gérer une cache mémoire que nous décrirons plus tard ainsi qu'une sorte de fenêtre ouverte sur une partie de cette cache permettant à l'utilisateur de pouvoir visualiser l'image.

6.4.1 La cache mémoire et la partie visible

L'utilisateur dispose sur son écran d'une fenêtre lui permettant de visualiser une partie de la mégaimage. Cette fenêtre est elle-même incluse dans une fenêtre plus grande qui est la cache mémoire. Lorsque l'utilisateur se déplace suivant l'axe X-Y, la fenêtre de visualisation se déplace dans cette cache mémoire.

La figure 6.5 illustre ces propos.

Ce procédé permet de répondre à un problème résultant de l'utilisation du paradigme client/serveur ainsi que de la décompression. En effet, comme nous l'avons déjà signalé, la décompression d'une partie d'image JPEG2000 n'est pas instantanée. Un certain temps de calculs relatifs à la puissance du processeur utilisé est nécessaire. De plus, de par le paradigme client/serveur, un temps de transfert est nécessaire pour le téléchargement de la partie d'image sur le serveur si celle-ci n'est pas en cache disque.

Ce procédé permet d'avoir toujours une certaine avance sur les déplacements de l'utilisateur pour peu que celui-ci se déplace de manière normale dans une mégaimage c'est-à-dire avec certains temps de pause permettant à la cache mémoire de se remplir complètement. Ce remplissage est réalisé par le module de décompression.

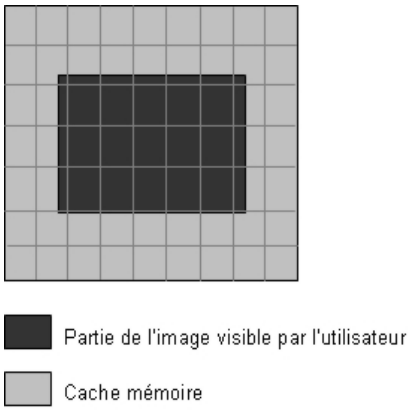

FIG. 6.5 – La cache mémoire et la partie visible

Il est important de signaler que cette cache s'accompagne d'une matrice représentant les tiles décompressées et de deux paramètres signalant les coordonnées X et Y de la tile de départ. Cette matrice évolue ainsi au fur et à mesure de la navigation. Comme nous le verrons plus tard, la matrice permettra à l'interpréteur de mouvements de connaître à tout instant l'ensemble des tiles qui doivent être décompressées pour disposer d'une image complète.

6.4.2 L'interpréteur de mouvement

Ce module a pour objectif d'interpréter les mouvements de l'utilisateur afin que celui-ci dispose toujours de l'information visuelle dont il a besoin. Chaque mouvement est interprété de manière à déplacer la fenêtre de visualisation et au besoin de demander au décompresseur de recharger certaines tiles.

Cinq types de mouvements différents peuvent être réalisés par l'utilisateur. Ceux-ci sont codifiés de la manière suivante :

- OPEN nom d'image : Lorsque l'utilisateur désire ouvrir une nouvelle image
- MOVEXY direction : Lorsque l'utilisateur désire se déplacer suivant l'axe X et Y dans l'image
- ZOOMIN niveau zoom : Signifie que l'utilisateur désire réaliser un zoom au niveau A
- ZOOMOUT niveau zoom : Signifie que l'utilisateur désire dézoomer au niveau B
- CLICK coordonnées X et Y : Précise que l'utilisateur vient de cliquer sur le slide view et que l'image principale doit être calibrée et rafraîchie sur les coordonnées X et Y.

La figure 6.6 illustre le mécanisme mis en place par l'interpréteur de mouvements lorsque l'utilisateur réalise des mouvements vers la droite.

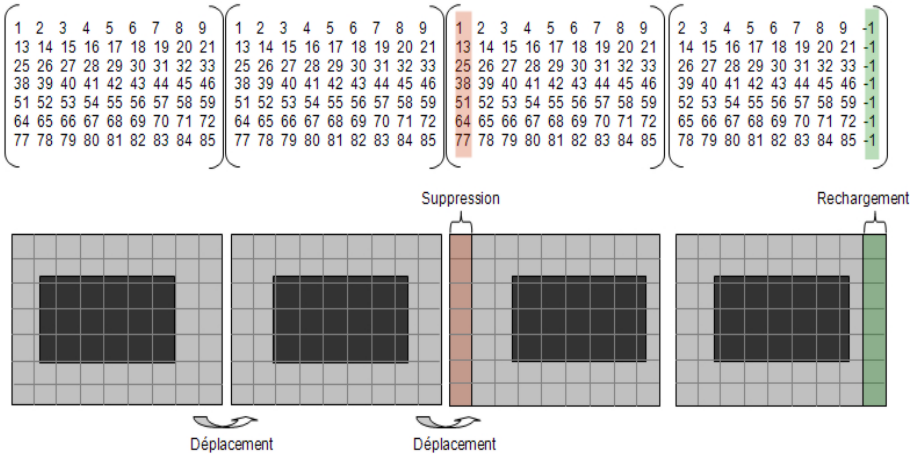

FIG. 6.6 – Déplacement cache mémoire

A chaque mouvement de l'utilisateur, l'interpréteur de mouvements vérifie la distance entre la fenêtre de visualisation et le bord de la cache mémoire. Si cette distance n'est pas inférieure à une certaine distance minimale paramétrée, l'interpréteur ne fait que déplacer de X pixels la fenêtre de visualisation dans la cache mémoire (ce que nous pouvons constater lors du premier déplacement sur le schéma 6.6).

Dans le cas contraire, le cadre de visualisation est déplacé et un ordre de rechargement des tiles du bord est lancé au décompresseur et la matrice de tiles est mise à jour. Grâce à cette matrice des tiles associée à la cache mémoire, l'interpréteur de mouvements peut savoir quelles tiles doivent être décompressées. Les tiles qui ne sont pas encore décompressées sont représentées par des -1. Dans le deuxième mouvement illustré sur le schéma 6.6, la matrice de tiles est décalée d'une colonne vers la gauche laissant ainsi apparaître des -1 sur toute la colonne de droite. Il est ainsi possible pour l'interpréteur de mouvements de déterminer les tiles à décompresser (ici 10, 22, 34, 47, 60, 73, 86) et ainsi d'envoyer l'ordre au décompresseur.

Pour l'interprétation des autres mouvements, le procédé est exactement le même. En se basant sur la matrice de positionnement et des coordonnées XY d'origine, il est possible de rafraîchir l'image. C'est pourquoi, nous n'entrerons pas plus dans les détails de ces autres mouvements.

6.4.3 Le décompresseur JPEG2000

Comme nous l'avons déjà signalé, à l'entrée de ce module, une file d'attente permet au décompresseur de réaliser les tâches qui lui sont demandées

par l'interpréteur de mouvements. Lorsque le décompresseur reçoit un ordre de décompression pour une tile, il doit disposer du code JPEG2000 de cette tile pour le décompresser. C'est pourquoi, il envoie un message à la cache disque afin d'obtenir le codestream correspondant. Il n'est donc pas du ressort du décompresseur de télécharger ce codestream. Pour lui, la cache disque doit être à même de lui fournir ce code.

Lorsque la cache disque lui fournit ce codestream, la tile est décompressée, mise à jour dans la cache mémoire et la matrice des tiles associée est également mise à jour pour signaler que la tile X vient d'être décompressée et affichée. Le décompresseur a donc réalisé la tâche qui lui était demandée et peut passer à la suivante.

6.4.4 Le module cache disque

Ce module a pour principale fonctionnalité de fournir au module de décompression les parties de codestream dont il a besoin. C'est donc ce module qui va être chargé de demander au serveur de lui fournir les tiles dans les résolutions adéquates. Au passage, ce module enregistre dans une cache disque les morceaux de codestream déjà téléchargés évitant ainsi de devoir les télécharger à nouveau si ceux-ci étaient éventuellement demandés plus tard. Pour plus de précisions, le lecteur peut consulter le diagramme d'état illustré au chapitre 4 de la partie "Matériel et méthode".

Dans une première approche, nous considèrerons que la cache dispose d'une place illimitée évitant ainsi de devoir gérer une politique de gestion de la cache[3]. Dans de futurs développements, nous pourrons concentrer nos efforts sur l'implémentation d'une politique adéquate.

6.5 Le protocole implémenté

Afin de permettre au client et au serveur de communiquer entre eux, un protocole propriétaire tel que nous en avions fait le choix dans le chapitre précédent a été développé. Les principes de base de ce protocole reposent sur des requêtes simples. La syntaxe de ce protocole est la suivante :

XXXX/(requête ou réponse) paramètres

Où XXXX est le code de la requête/réponse envoyée. Ce code permet de classer ces requêtes/réponses en différentes catégories.

- **00XX** : traitement des connexions/déconnexions
- **10XX** : demande d'informations sur les fichiers disponibles sur le serveur (liste, taille)

[3]Cfr. Matériel et méthode, Chap 4

– **20XX** : téléchargement des parties de fichiers JPEG2000 (en-têtes, résolutions de tiles)
– **99XX** : erreur (mauvais login ou mot de passe, mauvaise version du protocole, nombre maximum de connexions atteint sur le serveur)

Dans de futurs développements, de nouvelles requêtes et/ou catégories pourront au besoin être ajoutées.

6.6 Aperçu de l'application

6.6.1 Le serveur

Le serveur est présent sur une machine possédant les images au format JPEG2000. Les paramètres dont le serveur a besoin pour fonctionner sont : un numéro de port identifiant l'application sur le serveur, le nombre maximum de clients qui pourront se connecter simultanément, le répertoire contenant les mégaimages. La figure 6.7 illustre le lancement du serveur.

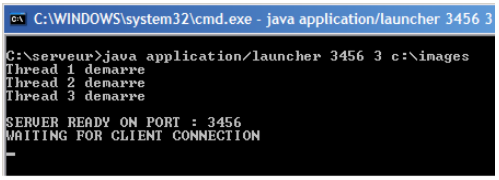

FIG. 6.7 – Lancement du serveur

Le serveur initialise le pool de thread avant de se déclarer prêt à recevoir les connexions des clients sur le port prévu à cet effet.

6.6.2 L'application cliente

IHM de connexion

Ce point présente l'IHM permettant de se connecter sur le serveur d'images à partir du poste client. Cette fenêtre est illustrée à la figure 6.8. Plusieurs champs sont présents sur cette fenêtre :

– Les champs login et mot de passe permettent à l'utilisateur de se faire reconnaître par le serveur.
– Le champs spécialité permet à l'utilisateur d'indiquer sa spécialité. Cette option n'est pas utilisée pour le moment mais pourra, dans des versions futures permettre une meilleure gestion de la cache disque. Nous y reviendrons dans le chapitre "Discussion".
– IP Server permet d'indiquer quel est l'IP du serveur sur lequel on désire se connecter.
– Server Port permet d'indiquer le port identifiant de l'application sur le serveur distant.

– Performance permet à l'utilisateur d'indiquer la performance de sa station de visualisation. Plus celle-ci est performante, plus la taille de la cache mémoire sera importante.
– Cache disk permet d'indiquer si l'on désire mettre en place un mécanisme de cache disque pour l'application.
– Cache location permet, si l'option cache disque a été sélectionnée, d'indiquer l'endroit où l'on désire stocker les éléments mis en cache.

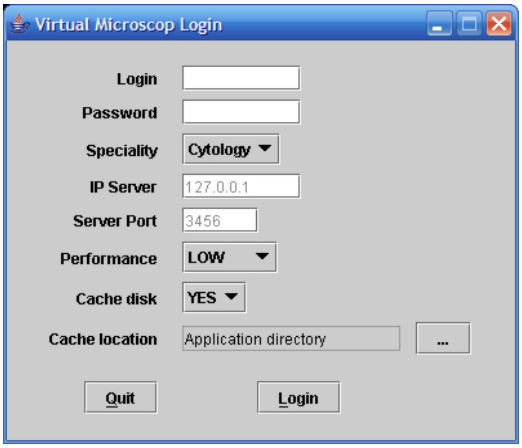

FIG. 6.8 – Fenêtre de connexion

IHM de la navigation

Dans ce point, nous allons détailler les caractéristiques de l'IHM de l'application cliente. Cette IHM est illustrée à la figure 6.9.

Au niveau des fonctionnalités, on retrouve sur l'IHM de l'application l'ensemble des fonctionnalités offertes par un microscope conventionnel. On y voit les déplacements suivant l'axe X-Y et également les changements d'objectifs. Une graduation permet également de réguler la vitesse de déplacement dans l'image. Cette fonction est utile lorsque l'utilisateur désire se déplacer à une vitesse plus lente dans les zones à analyser et de manière plus rapide pour atteindre d'autres zones.

Sur l'IHM, une barre d'outils est présente et propose d'ajouter un ensemble d'outils dont le spécialiste aura besoin pour l'établissement de son diagnostic. Dans la version actuelle de l'application, deux outils ont été implémentés afin de donner un aperçu des possibilités. A cet effet, un outil permettant de réaliser des captures d'écran a été développé ainsi qu'un outil permettant de comptabiliser les globules manuellement.

Dans un souci de navigation intuitive, le slide view permet à l'utilisateur de connaître sa position exacte dans l'échantillon. Ce slide view est également cliquable ce qui permet de se rendre directement à l'endroit désiré.

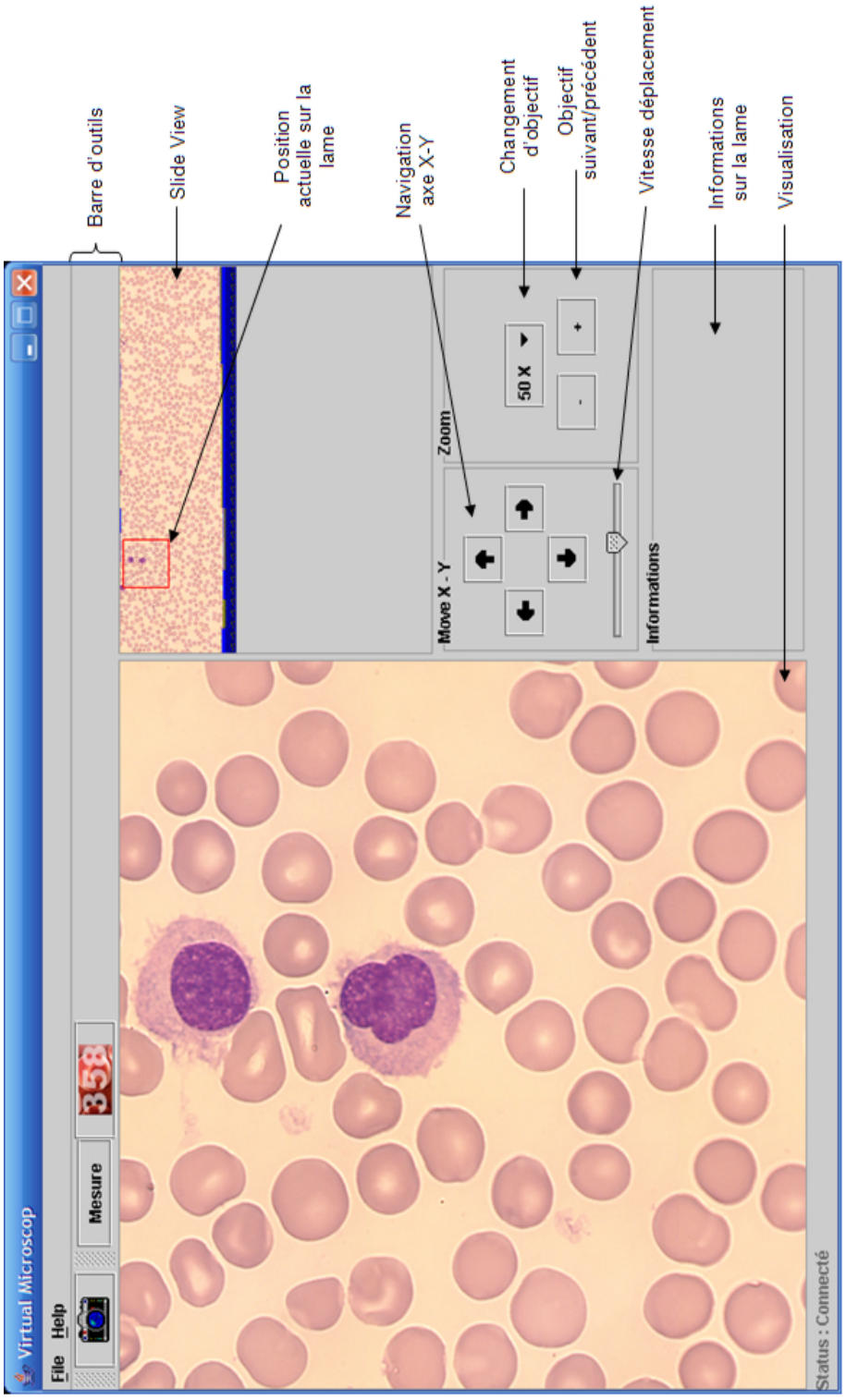

FIG. 6.9 – IHM du client

6.7 Conclusion

Ce chapitre nous a permis de mettre l'accent sur l'architecture développée tant du côté serveur que du côté client. Nous avons pu, dans cette architecture prendre en compte certains objectifs et choix réalisés dans le chapitre précédent. De par sa structuration en différents modules, nous pourrons, par la suite, améliorer l'application et y ajouter certaines fonctionnalités.

Cette application pourra dans de futures versions évoluer de manière à répondre à des objectifs plus précis mais également en ajoutant des fonctionnalités répondant de plus en plus aux nouvelles exigences des biologistes. Dans le dernier chapitre de ce travail, nous discuterons de différentes possibilités d'évolution et d'amélioration de l'architecture et de l'application en général.

Quatrième partie

Discussion

Chapitre 7

Discussion de la solution

7.1 Introduction

Dans la partie "Analyse et résultats" de ce travail, nous avons énuméré une série d'objectifs que l'application de visualisation se devait de respecter. Nous avons ensuite posé un ensemble de choix nous permettant de respecter ces objectifs lors du développement de cette application. Dans un souci de respect des délais impartis, certains choix ont été effectués afin de limiter le développement dans le temps. Il est donc important de souligner que certains de ces choix sont sujets à d'éventuelles évolutions.

Par la suite, nous avons présenté les caractéristiques de la solution développée. A cet effet, nous avons proposé une explication détaillée de l'architecture de l'application. Comme nous l'avons signalé, cette architecture présente une grande flexibilité et peut donc être sujette à d'éventuelles améliorations et extensions permettant de répondre à de futurs besoins.

Dans une optique de démarche scientifique, il peut être intéressant de poser une regard critique en prenant un certain recul vis-à-vis de la solution développée. Cette analyse permettra d'envisager certaines pistes d'amélioration et d'évolution pour l'application.

Dans ce chapitre, nous tenterons de vérifier si l'application développée est en totale adéquation avec les objectifs fixés. Nous présenterons ensuite les limitations de l'application existante et nous proposerons finalement quelques pistes d'évolution.

7.2 Réponse aux objectifs

Un accès à distance et sécurisé. L'application développée répond entièrement à cet objectif. Pour permettre l'accès à distance, l'architecture développée se décline en deux entités distinctes : un serveur et un client communiquant au travers un réseau de communication à l'aide d'un protocole de communication bien défini. Dans un souci de sécurité, lorsque le client

désire se connecter au serveur, le client doit se faire reconnaître à l'aide d'un login et d'un mot de passe. L'accès aux données n'est autorisé par le serveur que si ce login et ce mot de passe sont corrects.

Navigation intuitive et fluide. Il est très difficile de mesurer la pertinence de ces deux objectifs. La solution la mieux adaptée à l'évaluation de ces objectifs est de faire intervenir les personnes ayant l'habitude de réaliser des analyses à l'aide d'un microscope conventionnel. L'application fut donc testée par Mr. B Chatelain et par Mr. Y Cornet du laboratoire de Mont-Godinne afin de récolter leurs impressions sur les facteurs de fluidité et d'intuitivité de la navigation. Ceux-ci se sont avérés entièrement satisfaits par la qualité de la navigation offerte. Nous pouvons donc affirmer que l'application est totalement en phase avec les objectifs de fluidité et de navigation intuitive fixés.

La prise en compte de la diversité des environnements. Cet objectif est en grande partie réalisé. Grâce à l'utilisation du langage Java, il est en effet possible d'exécuter l'application sur des plateformes totalement différentes. Une nuance est à faire quant à l'adaptation aux paramètres de l'écran de l'utilisateur. Nous reviendrons sur cette critique dans la section traitant des limitations.

Les outils supplémentaires. L'application développée possède déjà quelques outils intéressants (captures d'images, compteur de globules). Afin de contrôler la qualité de ces outils, ceux-ci ont été testés par différentes personnes susceptibles d'utiliser l'application. Ces outils s'avèrent en totale adéquation avec les besoins des cytologistes pour les analyses. Une étude plus approfondie des outils nécessaires se doit d'être réalisée afin d'ajouter à l'application des outils supplémentaires utiles pour les biologistes.

Comme nous pouvons le constater, les objectifs fixés avant la réalisation de l'application ont été atteints. La grande possibilité d'évolution de l'application permet d'envisager, dans le futur, l'ajout de nouveaux objectifs où même l'affinement de certains d'entre eux par des techniques encore plus évoluées que celles utilisées.

7.3 Limite de l'application de visualisation

Avant d'envisager d'éventuelles évolutions, il peut être intéressant d'évoquer certaines limitations de l'application réalisée. Notons que ces limitations ne doivent pas être comprises comme des problèmes non résolus mais bien comme des fonctionnalités qu'il n'a pas été possible de développer dans les temps impartis. Ces limitations devront donc être levées dans de futurs développements.

7.3.1 Le créateur d'index

Comme nous l'avons déjà signalé, le créateur d'index est un logiciel qui a été développé par le laboratoire TELE de l'Université Catholique de Louvain (UCL). Ce programme permet au serveur de créer un fichier d'index associé à un fichier JPEG2000 lui permettant d'obtenir une structure du codestream. Ce logiciel a été écrit dans le langage C ce qui limite la portabilité du serveur. C'est pourquoi, dans les futures évolutions de l'application, il peut être intéressant de prévoir la réalisation d'une procédure Java permettant de réaliser cet index directement à partir du code source de l'application viewer.

7.3.2 Adaptation à l'écran de l'utilisateur

L'application actuelle se base sur l'option de performance sélectionnée sur la fenêtre de login pour déterminer la taille de la fenêtre de visualisation. Par manque de temps, il n'a pas été possible de développer une adaptation automatique à la résolution de l'utilisateur.

7.3.3 Amélioration du caching

Pour le développement de cette version de l'application, nous avions déjà précisé que nous n'inclurions pas de politique de gestion de la cache. Actuellement, l'application considère que la place disponible pour le stockage des données en cache est illimitée. Cependant, il faudrait améliorer ce point en incluant une politique de gestion de cache adaptée[1]. Il faudrait pouvoir tester chacune des différentes politiques existantes afin d'analyser laquelle correspondrait le mieux à la navigation de l'utilisateur.

7.4 Pistes d'évolution de l'application

Dans cette section, nous allons mettre en avant différentes possibilités d'évolution de l'application. Certaines permettront d'améliorer des caractéristiques déjà présentes comme la fluidité de la navigation, d'autres par contre seront des propositions d'ajouts de fonctionnalités. Dans ce point, nous nous contenterons d'évoquer certaines de ces pistes et d'en expliquer les grandes lignes. Il n'est pas dans l'optique de ce travail de réaliser une analyse approfondie des évolutions possibles, ce travail revenant plutôt à la personne désirant les développer.

Notons également que ces pistes ne sont que des propositions et que, pour la plupart, il est préférable d'en discuter avec les personnes concernées c'est-à-dire les biologistes afin de faire coïncider ces évolutions avec leurs attentes.

[1] Cfr. Matériel et méthode, Chapitre 4

7.4.1 Coopération à distance

Dans la version actuelle de l'application viewer, les résultats de navigation ne sont affichés que sur un seul poste de visualisation. Cependant, dans un domaine tel que l'analyse cytologique, il peut être intéressant de pouvoir partager la visualisation entre plusieurs personnes se trouvant à des endroits différents. Une personne possède les commandes de navigation et les autres visualisent en même temps sur leur écran. Cette évolution s'avère tout à fait possible grâce à la grande flexibilité de l'architecture développée.

La figure 7.1 illustre une évolution possible de l'architecture pour répondre à ce besoin. Dans un souci de visibilité du schéma, nous représentons seulement deux utilisateurs ensemble. Il est cependant tout à fait envisageable de gérer plus que deux utilisateurs en même temps.

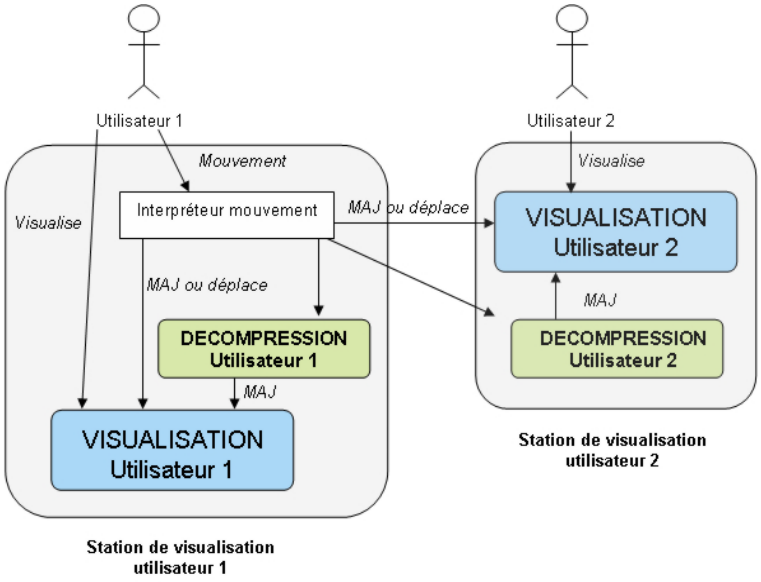

Fig. 7.1 – Coopération à distance

L'utilisateur principal peut naviguer dans l'image et faire profiter d'autres utilisateurs de sa navigation. Il est à noter que nous gardons toujours le principe de la décompression au niveau de chaque client.

Une autre évolution possible serait de permettre à tous les utilisateurs d'intervenir dans la navigation. Dans ce cas, il faudrait ajouter à l'architecture un module permettant de synchroniser les mouvements des différents utilisateurs.

7.4.2 Le préfetching

Comme nous l'avons déjà signalé dans le chapitre 4 de la partie Matériel et méthode, le préfetching consiste à remplir à l'avance la cache de manière à ce que, lorsque l'utilisateur a besoin de données, celles-ci se trouvent directement dans sa cache. On gagne alors un certain temps puisque l'on ne perd pas le temps de téléchargement de ces données.

On voit directement l'avantage du préfetching pour l'application de visualisation. Si l'on pouvait à l'avance mettre en cache les morceaux d'image susceptibles d'être visualisés par l'utilisateur dans un avenir proche, on pourrait améliorer la fluidité de la navigation en profitant des temps de pause de l'utilisateur pour les télécharger.

La difficulté du préfetching consiste en la réalisation d'un algorithme assez puissant pour pouvoir deviner quelles données doivent être téléchargées. Dans une application telle que celle réalisée dans ce travail, il faudrait pouvoir deviner à l'avance quels seront les mouvements réalisés par le spécialiste lors des prochaines actions.

La figure 7.2 tirée de [DOCM05] illustre ce principe.

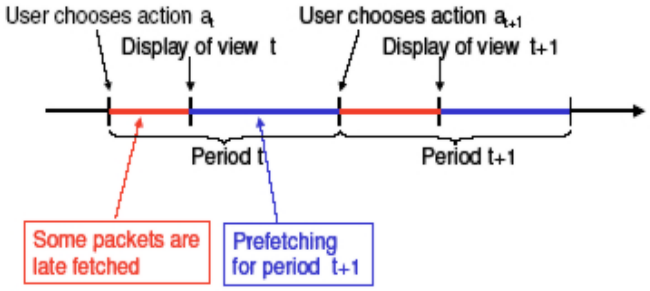

FIG. 7.2 – Prefetching

Pour tenter de résoudre ce problème, nous pouvons envisager trois pistes.

La première piste consiste à prendre comme hypothèse que dans une même spécialité, tous les individus observent l'image de la même façon[2]. Cette piste envisage donc de caractériser à l'avance le comportement de l'utilisateur lorsqu'il visualise une image. L'observation d'un biologiste particulier nous permettrait de dégager un comportement général. Ce comportement dégagé, il sera possible à chaque instant de prédire quelle sera la partie d'image qui sera visualisée lors du prochain mouvement. Après discussion avec certains biologistes, il s'avère qu'une pratique assez courante consiste à balayer l'image en Z de manière à n'oublier aucun globule dans l'analyse.

[2]D'où le champ spécialité présent sur la fenêtre de login

La figure 7.3 illustre ce balayage.

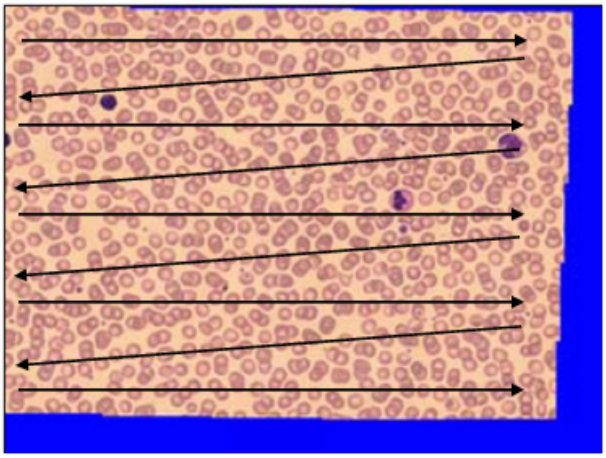

FIG. 7.3 – Visualisation cytologie

La deuxième piste consiste à prendre comme hypothèse que dans une même spécialité, tous les individus ne visualisent pas l'image de la même façon. Il est donc impossible de prédire à l'avance le comportement de l'utilisateur. La solution consisterait à analyser le comportement de l'utilisateur sur plusieurs images et d'en dégager une sorte de description du mouvement de l'individu. Ce qui par la suite, pourrait permettre de connaître à l'avance le comportement général de l'individu face à l'image.

La troisième piste ne tient pas compte de l'individu qui analyse l'image mais bien des caractéristiques propres à cette même image. Dans le domaine de la cytologie par exemple, les biologistes se déplacent dans l'image à un niveau de zoom très faible jusqu'à ce qu'ils trouvent un globule intéressant. A ce moment, un zoom est réalisé afin de visualiser plus précisément ce globule. En combinant des techniques de détection de globules dans les images, on pourrait détecter les globules dans l'image et prefetcher ces zones d'images en cache de manière à ce que, lorsque l'utilisateur désire réaliser un zoom sur un globule, celui-ci soit déjà dans sa cache. Outre la cytologie, il faudrait distinguer pour chaque domaine, quelles seraient les caractéristiques recherchées par les biologistes dans l'image pour les préfetcher.

La figure 7.4 illustre ce principe.

L'algorithme de préfetching pourra être intégré dans l'architecture de l'application de manière très aisée.

7.4.3 Aide à la décision

Le système de galerie d'images réalisé dans [Geo02] pourrait être combiné à cette application permettant ainsi de réaliser à partir de la mégaimage une

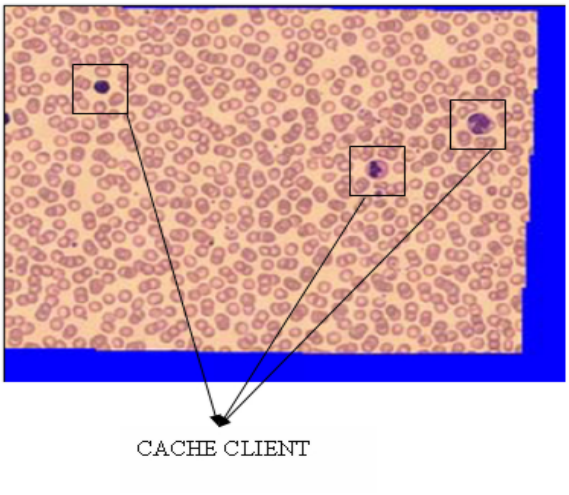

FIG. 7.4 – Prefetching zone

galerie d'images que le biologiste pourrait directement consulter.

En poussant encore l'analyse plus loin, on pourrait imaginer un logiciel permettant à partir d'une mégaimage, de poser lui-même un diagnostic que le biologiste pourrait confirmer en analysant lui-même l'image. L'objectif à atteindre serait la réalisation d'un logiciel d'aide au diagnostic extrêmement performant, s'appuyant sur une base de connaissance d'une très grande richesse.

Cette base de connaissances pourrait être une sorte de galerie d'images d'anomalies présentes dans le sang lors de certaines maladies. En comparant deux à deux les images, le logiciel pourrait détecter de manière automatique les anomalies et ainsi rendre un diagnostic.

On remarque directement l'inconvénient de ce genre d'application. La comparaison deux à deux des cellules entraînerait des temps de calculs beaucoup trop important ce qui rendrait le diagnostic beaucoup trop lent. C'est pourquoi une autre éventualité pourrait être de réaliser une étude permettant de caractériser les signes permettant de poser le diagnostic. Un regroupement en catégories permettrait d'aider plus facilement cette étude. De plus pour un diagnostic rapide et performant, il faudrait pouvoir diviser ces différentes catégories sous forme d'une arborescence de recherche affinant de plus en plus les critères de précision. L'ordinateur n'aurait plus qu'à parcourir l'arbre depuis la racine jusqu'à une feuille : cette feuille signalant les caractéristiques de la cellule que l'on analyse. Ce diagnostic automatique pourrait déjà permettre au biologiste d'avoir une orientation de son analyse pour lui faire gagner du temps.

La figure 7.5 illustre un exemple d'arbre de recherche simple basé sur la couleur, la forme et certains autres signes distinctifs d'une cellule. Nous

nous entendrons bien entendu sur le fait que ce genre de classification doit être mené à travers une étude très approfondie en collaboration avec des biologistes spécialistes du domaine.

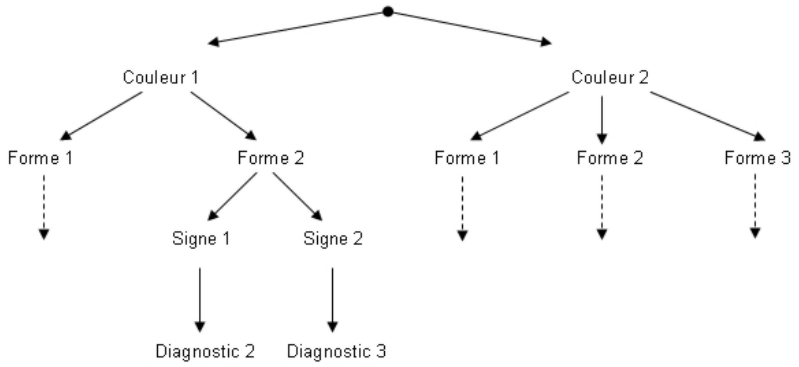

FIG. 7.5 – Arborescence pour le diagnostic automatique

7.4.4 Développement d'outils spécialisés

Cette piste pourra permettre aux biologistes de réaliser leur diagnostic avec encore plus de confort. L'aide à la décision citée précédemment peut être considérée comme un outil spécialisé. Cependant d'autres outils beaucoup moins complexes peuvent s'avérer très utiles.

- **Points d'intérêt.** Comme nous l'avons déjà précisé, les images visualisées sont de très grande taille. Lorsqu'un spécialiste réalise son diagnostic, il est parfois amené pour diverses raisons à devoir retourner à un endroit qu'il avait précédemment visualisé. C'est pourquoi, il faudrait pouvoir donner la possibilité au spécialiste de pouvoir retenir dans l'application certaines zones qu'il désirerait éventuellement revisualiser plus tard comme on le ferait avec le bookmark d'un browser internet pour une page html.
- **Ajout d'annotations et de commentaires.** Cette fonctionnalité permet au biologiste d'annoter directement l'image avec un ensemble de symboles mis à sa disposition (flèches, couleurs, lignes, cercles, commentaires,...). Pour pouvoir sauvegarder ces annotations pour des sessions futures, deux possibilités sont offertes : soit associer au fichier image un fichier listant l'ensemble des annotations réalisées par l'utilisateur, soit profiter des caractéristiques de JPEG2000 offrant la possibilité d'ajouter des tags dans les en-têtes JPEG2000. La première solution sera privilégiée si les annotations réalisées sont différentes pour chaque personne désirant visualiser l'image. Pour des annotations communes, la deuxième solution convient parfaitement.

- **Réaliser des mesures sur l'échantillon.** Il peut être intéressant de réaliser un petit outil permettant au biologiste en deux clics sur l'image de connaître la distance entre deux régions particulières. A cet effet, on s'interrogera sur la distance réelle représentée à l'écran.
- **Travail sur les couleurs de l'image.** On pourra par exemple penser à des fonctionnalités comme un travail sur la luminosité, sur les contrastes de l'image ou même sur les niveaux de couleurs. Ceci permettra à l'utilisateur de faire ressortir certains détails de l'image qui ne seraient pas facilement observables sans ces fonctionnalités.
- **Autres...** C'est au fur et à mesure de l'utilisation que les biologistes se rendront compte qu'il serait intéressant de développer tel ou tel outil supplémentaire.

7.5 Conclusion

L'application réalisée répond donc de manière satisfaisante aux objectifs fixés. Comme nous l'avons signalé dans ce chapitre, des améliorations sont encore possibles et l'ajout de fonctionnalités est tout à fait possible. Nous en avons déjà évoqué quelques-unes. Bien entendu, bon nombre d'autres extensions sont encore envisageables. C'est en discutant avec les principaux utilisateurs de l'application que les besoins seront mis en lumière.

Conclusion

La télémicroscopie est une discipline en constante évolution. A l'heure actuelle, beaucoup de projets de recherche sont menés dans ce domaine. Parmi ceux-ci, le projet de télémicroscopie du laboratoire de cytologie de l'UCL Mont-Godinne lancé quelques années auparavant, étudie la réalisation d'une plateforme de télémicroscopie "store and forward" permettant à des experts éloignés de réaliser leur diagnostic à distance. Depuis plusieurs années, plusieurs travaux de recherche se sont déjà succédé dans la réalisation de ce projet.

C'est dans la poursuite de ces différents travaux que ce mémoire s'est inscrit. Son objectif était la réalisation d'une application de visualisation client-serveur permettant à un biologiste de consulter, grâce à son ordinateur, les mégaimages précédemment capturées et ce, à la manière d'un microscope conventionnel. La difficulté résidait justement dans cette optique d'imiter du mieux possible l'aspect de fluidité du microscope conventionnel.

A travers ce travail, nous avons analysé toute une série de pistes nous permettant de réaliser une application la plus confortable et la plus efficace possible. A cet effet, un état de l'art du domaine nous a permis de connaître les dernières évolutions en matière de télémicroscopie. Par la suite, nous avons tenté de rechercher les techniques les plus adéquates pour la réalisation de cette application. Nous avons poursuivi par la définition de quelques grands objectifs à satisfaire pour qu'une telle application soit utilisable dans le domaine médical. Finalement, une recherche en matière d'architecture a permis à l'application de fonctionner de manière optimale en intégrant les techniques découvertes.

Le resultat de ce travail a mené à l'élaboration d'une puissante application de visualisation menant à un outil utilisable dans le domaine de la télémicroscopie. La fluidité et le confort offerts par la navigation dans les mégaimages est le résultat de la recherche approfondie menée à travers ce mémoire.

Bien que l'application développée réponde aux objectifs fixés avec les

différents spécialistes, elle est toujours susceptible d'être améliorée. Dans la dernière partie de ce mémoire, nous avons évoqué quelques pistes d'évolution pour l'application. Par des techniques de prefetching, la navigation peut s'avérer encore plus performante. De plus, l'ajout d'outils nécessaires au diagnostic doit également être pris en compte dans les objectifs d'évolution.

Il est important de noter que pour de futures évolutions de l'application, il ne faut pas oublier d'intégrer dès le départ les principaux utilisateurs de l'application à savoir les biologistes du laboratoire. C'est en récoltant leurs impressions et leurs idées qu'il sera possible de développer une application leur permettant d'exercer leur savoir-faire le plus efficacement possible.

Pour terminer, nous pouvons affirmer que l'UCL Mont-Godinne dispose à l'heure actuelle d'une plateforme de télémicroscopie utilisable et que son avance dans ce domaine amène le laboratoire de cytolgogie au rang de précurseur en matière de télémicroscopie en cytologie.

Bibliographie

[AWA] AWARE. Jpip software toolkit. `http://sic.epfl.ch/SA/publications/FI01/fi-3-1/3-1-page1.html` (Dernière visite : 22 janvier 2005).

[Boc] Vincent Bockaert. Interpolation. `http://www.dpreview.com/learn/?/Glossary/Digital_Imaging/Interpolation_01.htm` (Dernière visite : 14 mai 2005).

[CDM+02] J-C. Chevrolet, M. Denz, B. Merminod, S. Osswald, and M. Roulet. *Télémédecine*. Académie suisse des sciences techniques SATW, 2002. ISBN 3-908235-07-3.

[COM00] JPEG2000 COMMITTEE. Jpeg 2000 part i final committee draft version 1.0. page 205, Mars 2000. `http://www.jpeg.org/public/fcd15444-1.pdf` (Dernière visite : 21 janvier 2005).

[CP02] M. CONTOUX and F. PRODHOMME. Etat de l'art de l'apport de la télémédecine en hospitalisation à domicile. `http://www.utc.fr/~farges/DESS_TBH/01-02/Projets/` (Projet DESS "TBH", UTC), 2002.

[CS99] C. Christopoulos and A. Skodras. Jpeg 2000 - the next generation still-image compression standard. Octobre 1999. Tutorial given at the IEEE International Conference on Image Processing (ICIP).

[DD03] A. Descampe and F-O. Devaux. Etude et conception d'un décodeur hardware jpeg 2000 destiné au cinéma numérique. Mémoire de fin d'études. UCL(Université Catholique de Louvain) Belgium, JUIN 2003.

[Dec04] G. Decock. Composition automatique de frottis numériques - application au gigaimages. Mémoire de fin d'études, FUNDP(Facultés Universitaires Notre-Dame de la Paix) Namur Belgium, Août 2004.

[DOCM05] Antonin Descampe, Jihong Ou, Philippe Chevalier, and Benoit Macq. Data prefetching for smooth navigation of large scale jpeg2000 images. 2005.

[DS01] T. Ebrahimi D. Santacru, R.Grosbois. Jpeg 2000 - la nouvelle norme pour le codage d'images, 2001. http://sic.epfl.ch/SA/publications/FI01/fi-3-1/3-1-page1.html (Dernière mise à jour : 27 mars 2001) (Dernière visite : 16 janvier 2005).

[Geo02] B. Georges. La télémicroscopie en cytologie hématologie. Mémoire de fin d'études, FUNDP(Facultés Universitaires Notre-Dame de la Paix) Namur Belgium, Août 2002.

[Gla] Randy Glass. What is interpolation ? http://home.surewest.net/frcn/Interpolation.html (Dernière visite : 14 mai 2005).

[Gra00] Karen L. Gray. The jpeg2000 standard. (1), Janvier 2000.

[Int] InterpolateTHIS.com. What is interpolation ? http://www.interpolatethis.com/interp.html (Dernière visite : 14 mai 2005).

[Leo] F. Joel Leong. What is telepathology ? http://www.telepathologycity.com/what_is_telepathology.htm (Dernière visite : 4 mars 2005) Mirada Solutions, an Oxford University spinoff company.

[Lur] Luratech. The jpeg 2000 source. http://www.jpeg2000info.com/where/ (Dernière visite : 16 janvier 2005).

[MA] F. De Montbel and Al. Télémédecine et imagerie médicale. http://www.ese-metz.fr/metz/eleves/themes/imagerie/tele_part1.html (Dernière visite : 19 fevrier 2005).

[MI] F-O. Devaux M. Iregui, A. Descampe. Jpeg 2000. Présentation Power Point réalisée à l'UCL (Université Catholique de Louvain), Belgium.

[MPS+] F. Midy, D. Polton, A. Strauss, F. Kletz, JC. Moisdon, and C. Kornblum. Télémédecine et Évaluation. http://www.sante.gouv.fr/htm/dossiers/telemed/tele_eval/sommaire.htm (Dernière visite : 12 fevrier 2005).

[Rap] Raphael.Grosbois. Jj2000 - an implementation of the jpeg2000 standard in java. http://jj2000.epfl.ch/ (Dernière visite : 30 mars 2005).

[RD] France Télécom RD. Télémédecine. http://www.rd.francetelecom.com/fr/galerie/telemedecine/pdf/doc.pdf (Dernière visite : 23 fevrier 2005).

[Ren03] S. Renard. Première approche de jpeg2000. *Club photoshop*, page 11, Septembre 2003.

[Sca] Scanscope. Scanscope - welcome to the world of virtual slide technology. http://www.scanscope.com/scanscope/ (Dernière visite : 26 fevrier 2005).

[Tau02] D. Taubman. Jpip - jpeg2000 internet protocol in kakadu v3.3. http://www.kakadusoftware.com/jpip-kakadu-superceded.pdf (Dernière visite : 22 janvier 2005), Aout 2002.

[Tau03] D. Taubman. Jpeg2000 compression standard for interactive imaging. Présentation PowerPoint, Mars 2003.

[TP] D. Taubman and R. Prandolini. Architecture, philosophy and performance of jpip : Internet protocol standard for jpeg2000. http://www.ee.unsw.edu.au/~taubman/publications_files/JPIP-architecture-vcip03.pdf (Dernière visite : 22 janvier 2005).

[Tre] Trestle. Medmicroscopy - internet microscopy solution. http://www.trestlecorp.com/corp_dllit/Broch_MM_4pg.pdf (Dernière visite : 25 fevrier 2005).

[Zei] Zeiss. Axiopath - the telemicroscopy system. http://www.zeiss.de/de/micro/home_e.nsf/InhaltWWWIntern/0297613BE11435F44125687A00575555 (Dernière visite : 25 fevrier 2005).

[Zuy03] L. Zuyderhoff. Composition automatique de frottis numériques. Mémoire de fin d'études, FUNDP(Facultés Universitaires Notre-Dame de la Paix) Namur Belgium, Août 2003.

Annexe A

Le fichier d'index

Dans cette annexe, nous allons présenter la syntaxe du fichier d'index associé au codestream JPEG2000. Cet index permet au serveur de connaître la structure exacte du codestream. Nous présenterons ensuite un exemple de fichier d'index.

A.1 La syntaxe

<Largeur Image> <Hauteur Image>
<Ordre de progression (LRCP)>
<Largeur d'une tile> <Hauteur d'une tile>
<Nombre de tiles en largeur> <Nombre de tiles en hauteur>
<Nombre de components>
<Nombre de décompositions en ondelettes>
<Taille des precincts à chaque résolution>
<Index du dernier byte du Main Header>
<Taille du codestream en bytes>
<Num de tile> <resolution> <byte départ> <byte fin>
....

Nb : Une résolution "-1" signifie que les bytes indiqués représentent l'en-tête de la tile.

A.2 Exemple de fichier d'index

Voici l'exemple d'un fichier d'index généré pour une image JPEG2000 de 539 tiles.

6912 5120
1
256 256
27 20
3
5
[32768,32768][32768,32768][32768,32768][32768,32768][32768,32768]
134
4184700
0 -1 135 199
0 0 200 435
0 1 436 859
0 2 860 1579
0 3 1580 2763
0 4 2764 5339
0 5 5340 7680
1 -1 7681 7745
1 0 7746 7984
1 1 7985 8426
1 2 8427 9183
1 3 9184 10500
1 4 10501 13170
.....
539 -1 4184572 4184636
539 0 4184637 4184667
539 1 4184668 4184673
539 2 4184674 4184679
539 3 4184680 4184685
539 4 4184686 4184691
539 5 4184692 4184697

Annexe B

L'interpolation de pixels

Cette annexe est un condensé des références [Int, Gla, Boc]. Dans cette annexe, nous tenterons d'énoncer les principes fondamentaux de l'interpolation de pixels. A travers différentes explications et illustrations, nous présenterons un éventail de différentes techniques existantes. Bien que beaucoup de techniques existent, celles présentées ci-dessous sont parmi les plus utilisées dans les logiciels d'images.

B.1 L'interpolation

L'interpolation de pixels est une méthode utilisée pour augmenter la taille d'une image de manière mathématique. C'est une manière d'estimer la valeur des pixels laissés libres par l'augmentation de la surface. On comprend que cette opération donne d'assez bons résultats si le nombre de pixels à interpoler n'est pas trop important et si l'image est surtout composée de grandes zones uniformes. La figure B.1 illustre à gauche le fragment d'une image agrandie sans interpolation, à droite l'agrandissement de ce même fragment en utilisant une technique d'interpolation quelconque.

FIG. B.1 – Image avec et sans interpolation

Pour illustrer nos explications, prenons l'exemple d'une image en niveau de gris c'est-à-dire dont chaque pixel est représenté par une seule valeur (entre

0 et 255). Cette image d'une dimension de 400 x 300 pixels doit être élargie à 1600 x 1200 pixels. Cela signifie que d'une image d'origine de 120.000 pixels, on obtient une image élargie de 2 millions de pixels. L'ordinateur va donc devoir estimer la valeur des nouveaux pixels en se basant uniquement sur les 120.000 valeurs originales. La figure B.2 illustre ces propos.

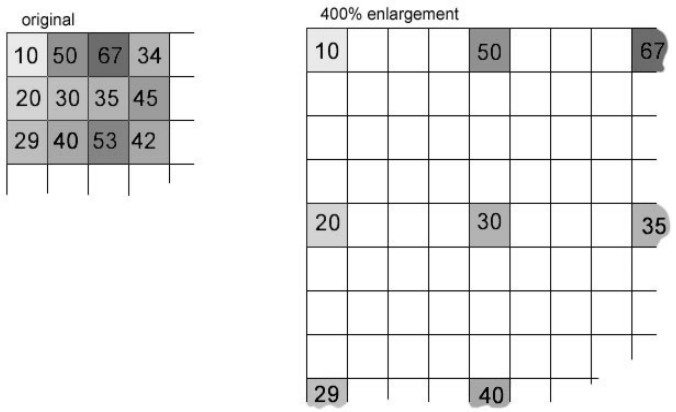

FIG. B.2 – Illustration de l'interpolation

La figure B.3 servira d'exemple pour illustrer les résultats de l'application des différentes méthodes d'interpolation présentées.

FIG. B.3 – Image de départ

B.2 Techniques d'interpolation

Différentes techniques d'interpolation de pixels existent. Certaines offrant de meilleurs résultats que d'autres ou produisant des résultats beaucoup plus rapidement. Pour choisir une technique d'interpolation pour une application il convient d'évaluer celle-ci en fonction de deux paramètres : le qualité de l'image à produire et la rapidité de production de cette image.

B.2.1 La méthode "le voisin le plus proche"

Cette méthode d'interpolation est la plus simple et la plus rapide. Le principe de base consiste à rendre les pixels plus grands. La couleur d'un

pixel dans la nouvelle image est la couleur du pixel le plus proche dans l'image originale. Si l'on agrandit de 400% l'image originale, cela signifie qu'un pixel sera répété quatre fois dans chaque direction (horizontale et verticale). La plupart des logiciels de visualisation et d'édition d'images emploient cette technique d'interpolation pour agrandir les images numériques parce qu'elle ne change pas l'information de couleur de l'image et ne présente aucun anti-aliasing[1]. La figure B.4 illustre ce principe.

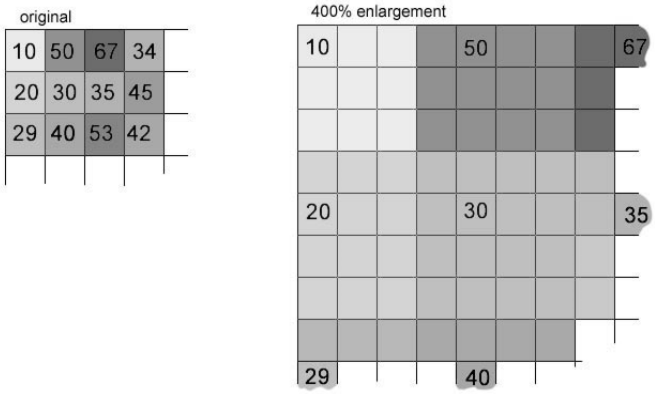

FIG. B.4 – Illustration de l'interpolation le "voisin le plus proche"

La figure B.5 illustre l'agrandissement de l'image de départ par la technique d'interpolation "le voisin le plus proche".

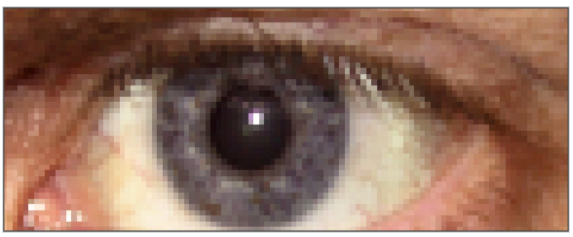

FIG. B.5 – Illustration de l'interpolation "le voisin le plus proche" sur l'exemple de départ

B.2.2 L'interpolation bilinéaire

Cette méthode produit des résultats de qualité moyenne. Cette technique d'interpolation consiste à effectuer la moyenne des quatre pixels les plus

[1]Technique par laquelle on diminue l'effet d'escalier des images, en créant des dégradés de couleurs le long des contours, pour les lisser

proches dans l'image originale. La figure B.6 illustre ce principe.

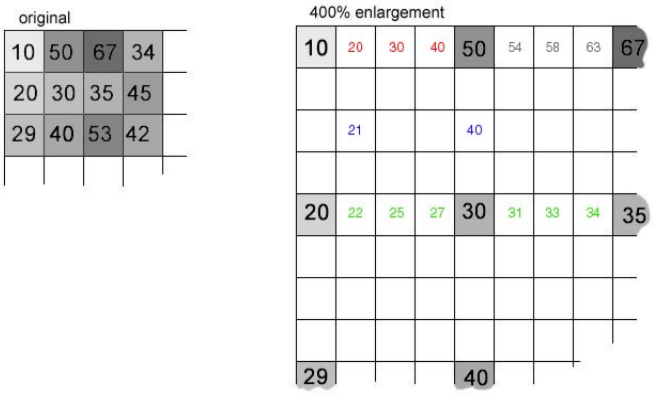

FIG. B.6 – Illustration de l'interpolation bilinéaire

La figure B.7 illustre l'agrandissement de l'image de départ par la technique d'interpolation bilinéaire.

FIG. B.7 – Illustration de l'interpolation bilinéaire sur l'exemple de départ

B.2.3 L'interpolation bicubique

L'interpolation bicubique est plus sophistiquée et produit des résultats encore meilleurs que l'interpolation bilinéaire. Cette technique évalue un nouveau pixel par une fonction utilisant les 16 pixels les plus proches dans l'image d'origine. Cette méthode est celle la plus utilisée pour les drivers d'impression et les méthodes de rééchantillonnage des appareils photos digitaux. Cependant, elle est beaucoup plus lente que les précédentes car celle-ci prend en compte un nombre de pixels plus important.

La figure B.8 illustre l'agrandissement de l'image de départ par la technique d'interpolation bicubique.

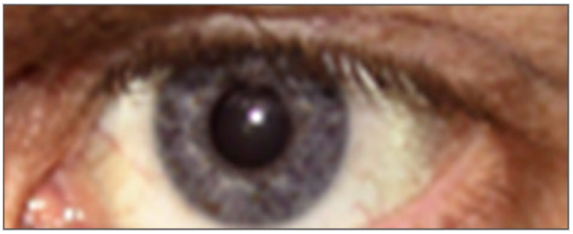

FIG. B.8 – Illustration de l'interpolation bicubique sur l'exemple de départ

B.2.4 L'interpolation fractale

Cette méthode est principalement utilisée pour de grands agrandissements. Les résultats obtenus sont encore meilleurs que ceux obtenus par l'interpolation bicubique. Cette technique emploie des modèles mathématiques sophistiqués pour détecter les bords des formes et garder les parties pointues de l'image. Ainsi, les formes des objets sont encore plus précises et l'effet de halos autour des bords est atténué. Cette technique offrant des résultats très satisfaisants demande cependant des temps de calcul assez important et n'est pas adaptée à des applications exigeant un zoom quasi instantané.
La figure B.9 illustre l'agrandissement de l'image de départ par la technique d'interpolation fractale.

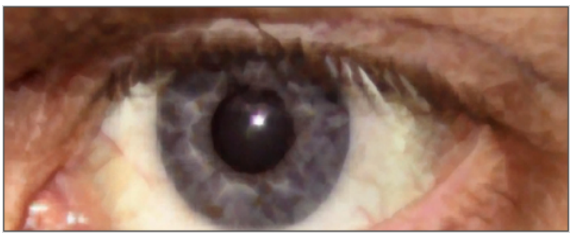

FIG. B.9 – Illustration de l'interpolation fractale sur l'exemple de départ